Asynchronous Federated Learning Over Non-IID Data via Over-the-Air Computation

Qianpiao Ma, Xiaozhu Song, Junlong Zhou, *Member, IEEE*, Haibo Wang, *Member, IEEE*, Yunming Liao, Jianchun Liu, *Member, IEEE*, and Hongli Xu, *Member, IEEE*

***Abstract*—Federated learning (FL) enables training AI models across distributed edge devices (*i.e.*, workers) using local data, while facing challenges including communication resource constraints, edge heterogeneity, and non-IID data. Over-the-air computation (AirComp) has emerged as a promising technique to improve communication efficiency by leveraging the superposition property of a wireless multiple access channel (MAC) for model aggregation. However, over-the-air aggregation requires strict synchronization among edge devices, which is essentially incompatible with the asynchronous FL mechanisms often used to handle edge heterogeneity. To overcome this incompatibility, we propose Air-FedGA, a grouping-based asynchronous FL mechanism via AirComp, where workers are organized into groups for synchronized over-the-air aggregation within each group, while groups asynchronously communicate with the parameter server to update the global model. This design retains the communication efficiency of AirComp while addressing training inefficiency caused by edge heterogeneity. We provide a rigorous convergence analysis for Air-FedGA, theoretically quantifying how the convergence bound depends on several key factors, such as the maximum staleness, the degree of non-IID data among groups, and the AirComp aggregation mean squared error (MSE). Guided by these theoretical insights, we propose power control and worker grouping algorithms to minimize the convergence bound by jointly optimizing the AirComp aggregation MSE and the grouping strategy. We conduct experiments on classical models and datasets, and the results demonstrate that our proposed mechanism and algorithms can accelerate the model training by 1.83-2.22× compared with the state-of-the-art solutions.**

***Index Terms*—Edge Computing, federated learning, Over-the-air computation, asynchronous, heterogeneity, Non-IID.**

I. Introduction

THE rapid growth of the Internet of Things (IoT) has led to the generation of massive amounts of data from edge devices, such as sensors, mobile phones, and base stations [2]. Effectively utilizing this data is crucial for enhancing the quality of service (QoS) on various applications, such as interactive online gaming, face recognition, 3D modeling, VR/AR and vehicle networking systems. Recently, artificial intelligence (AI) algorithms have deployed from the centralized cloud to the distributed network edge, which is known as edge AI [3], enabling efficient data processing locally. This shift has facilitated the adoption of federated learning (FL), a technique that trains AI models over edge nodes while preserving privacy by utilizing data locally.

FL has gained significant attention since its introduction in 2016 [4]. A typical FL system usually consists of a large number of edge devices (*i.e.*, workers) for local model training, and a centralized parameter server (PS) for each round of local model aggregation. Each worker trains model over its local dataset and sends the trained local model to the parameter server. The parameter server aggregates these local models into a global model and broadcasts it back to the workers. This procedure continues for multiple rounds until the global model converges.

However, when deployed at the network edge, FL faces several challenges: **1) Limited communication resource**: Traditional orthogonal multiple access (OMA) schemes (TDMA [5], [6], [7], OFDMA [8], [9]) are commonly used for FL model aggregations. However, due to limited communication resources, the transmission delay increases linearly with the number of workers when deploying OMA schemes [10], resulting in unsatisfactory scalability in large-scale FL scenarios. **2) Edge heterogeneity**: The CPU capacities, data sizes, and network connections are usually heterogeneous at network edge. As a result, the required time to perform local updating and receive/upload models may vary significantly. **3) Non-IID Data**: A device's local data are often not sampled drawn uniformly from the overall distribution. In other words, the

Received 9 July 2025; revised 4 November 2025; accepted 5 December 2025; approved by IEEE TRANSACTIONS ON NETWORKING Editor E. E. Tsiropoulou. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 62402537, Grant U25A20442, Grant 62572244, and Grant 62172224; in part by the Fundamental Research Funds for the Central Universities under Grant 30925010408; in part by the Natural Science Foundation of Jiangsu Province under Grant BK20220138; and in part by the Open Research Fund of the National Mobile Communications Research Laboratory, Southeast University, under Grant 2024D07. Some earlier results of this paper were published in the Proceedings of IEEE International Parallel & Distributed Processing Symposium (IPDPS), 2025 [1]. *(Corresponding authors: Junlong Zhou; Jianchun Liu.)*

Qianpiao Ma and Xiaozhu Song are with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, Jiangsu 210094, China (e-mail: maqianpiao@njust.edu.cn; songxiaozhu@njust.edu.cn).

Junlong Zhou is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, Jiangsu 210094, China, and also with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 211111, China (e-mail: jlzhou@njust.edu.cn).

Haibo Wang is with the Department of Computer Science, University of Kentucky, Lexington, KY 40506 USA (e-mail: haibo@ieee.org).

Yunming Liao, Jianchun Liu, and Hongli Xu are with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China, and also with Suzhou Institute for Advanced Study, University of Science and Technology of China, Suzhou, Jiangsu 215123, China (e-mail: ymliao98@mail.ustc.edu.cn; jcliu17@ustc.edu.cn; xuhongli@ustc.edu.cn).

Digital Object Identifier 10.1109/TON.2025.3641928

TABLE I
PERFORMANCE COMPARISON FOR FL MECHANISM

FL Mechanism	Communication Consumption	Scalability	Handing Edge Heterogeneity	Handing Non-IID
Synchronous[4–9, 11]	Medium	Poor	Poor	Medium
AirComp+Synchronous[10, 14–21]	Low	Poor	Poor	Medium
Asynchronous[22–30]	High	Good	Good	Poor
AirComp+Asynchronous [31, 32]	Low	Good	Good	Poor
AirComp+Asynchronous (**Air-FedGA**)	Low	Good	Good	Good

local data on edge devices are typically non-independent and non-identically distributed (non-IID) [11].

Traditional OMA schemes aggregate models on the premise that all model vectors can be reliably transmitted. However, the parameter server actually just needs to obtain the weighted average of the local model vectors, rather than ensuring reliable transmission of each individual vector [12]. To this end, *over-the-air computation* (AirComp) [13], a new analog non-orthogonal multiple access (NOMA) technique, is motivated to break through the limitation of communication resources, and reduce transmission delay for large-scale FL [14], [15], [16], [17], [18], [19]. AirComp-based aggregation is achieved by synchronizing workers to transmit their local model vectors concurrently, leveraging the superposition property of wireless multiple access channels (MAC) to sum these vectors over-the-air [14].

The implementation of AirComp has a bifacial impact on FL. On the one hand, due to the efficient spectrum utilization of AirComp-based aggregation, it is expected to significantly reduce the transmission delay compared to the OMA-based aggregation which decouples communication and computation. On the other hand, strict synchronization among all heterogeneous workers is required for successful over-the-air aggregation [10]. However, due to edge heterogeneity, the completion time of workers varies significantly [23]. As a result, the PS has to wait for the slowest worker to complete its local training before the over-the-air aggregation, while other workers are idle at this time. It is known as the *straggler problem* [22], leading long single-round duration and poor scalability. Conventionally, asynchronous FL mechanisms are deployed to tackle the straggler problem [22], [23], [24], [25], [26], [27], [28], [29], [30]. In this case, the PS updates the global model with each local model as soon as it arrives, without waiting for others. However, a fully asynchronous mechanism *is incompatible with AirComp-based aggregation*, since the implementation of AirComp is based on the concurrent transmission of multiple devices.

To this end, several recent studies [31], [32] propose time-triggered asynchronous FL mechanisms, where each worker pauses after completing local training until a predefined synchronization point, at which all waited workers perform over-the-air aggregation simultaneously. However, by performing aggregation only on the basis of synchronization point, these approaches ignore non-IID data among workers. In fact, the data distribution among workers participating in each aggregation significantly impacts the training performance [33].

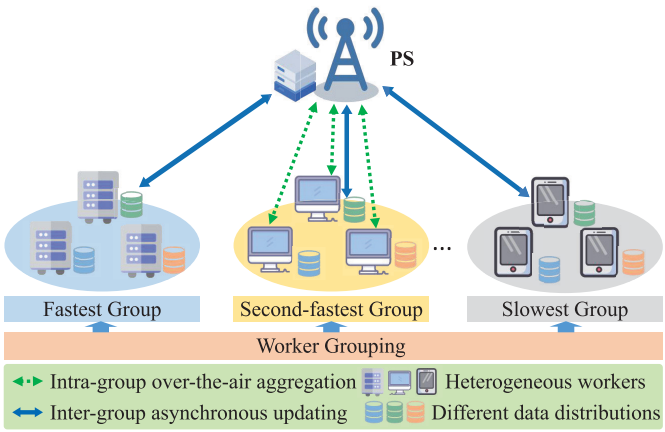


Fig. 1. The architecture of Air-FedGA.

In this paper, we propose a grouping-based asynchronous federated learning mechanism via over-the-air computation (Air-FedGA). As illustrated in Fig. 1, the workers are organized into groups considering both edge heterogeneity and data distribution. Air-FedGA accelerates federated learning by relaxing the strict synchronization requirement of AirComp: workers within each group perform over-the-air aggregation simultaneously, while global updating occur asynchronously across groups. Unlike our previous work [1], which employed a heuristic algorithm for worker grouping without performance guarantees, this paper reformulates the worker grouping as a convergence-aware problem and then transforms it into a convex optimization problem. Solving this program yields grouping strategy that ensure workers within each group have similar local training times to address edge heterogeneity, while promoting inter-group data distributions as close to IID as possible to mitigate the adverse effects of non-IID data. More details of the improvements of this work compared to [1] is presented in Section V. The comparison of different FL mechanisms are summarized in Table I.

The main contributions of this paper are as follows:

- We design Air-FedGA, a mechanism that overcomes the limitations of deploying asynchronous FL via over-the-air computation, to address the communication constraints and edge heterogeneity. We analyze the convergence of Air-FedGA, and explore the quantitative relationship between the convergence bound and key factors, *e.g.*, the maximum staleness, the degree of non-IID data among groups, and the AirComp aggregation mean squared error (MSE).

- Based on the analysis, we formulate a convergence bound minimization problem given the training time and energy constraints. To solve it, we first decouple and jointly optimize the power scaling factors at workers and the denoising factors at parameter server. Next, we transform the original problem into a convex optimization problem using the quadratic transform and relaxation techniques. Finally, we design a rounding-based worker grouping algorithm to solve the problem and derive its approximate ratio.
- Experimental results on the classical models and datasets show that our proposed mechanism and algorithms can greatly accelerate the model training by 1.76-2.16× compared with the state-of-the-art solutions.

The rest of this paper is organized as follows. Section II reviews the related works. Section III introduces the grouping-based asynchronous federated learning via AirComp, and Section IV provides its convergence analysis. The worker grouping algorithm is proposed in Section V. The experimental results are shown in Section VI. Section VII concludes this paper.

II. Related Work

A. Asynchronous Federated Learning

Asynchronous federated learning (FL) addresses the straggler problem by aggregating a local model with the global model as soon as the parameter server receives it, without waiting for all workers to finish their local training. However, this comes at the expense of using out-of-date models, inevitably incurring *staleness* concern [22].

Many researches have focused on addressing edge heterogeneity with asynchronous scheme while mitigating the adverse effects of staleness. For example, Xie et al. [22] assign smaller aggregation weights to stale models to lessen their impact on training. Wu et al. [23] and Chen et al. [24] propose simple approaches to handle staleness, where the parameter server discards too stale models during training process. Chen et al. [25] deploy dynamic learning rates for workers according to the frequency of their participating in global updating, which can also alleviate the staleness concern. Zheng et al. [26] and Zhu et al. [27] compensate delayed gradients based on approximate Taylor expansion. Ma et al. [28] introduce a semi-asynchronous FL mechanism that involves multiple workers in each global updating to ensure that the model is not too stale. Chai et al. [29] organize workers into groups according to their communication time with the parameter server, and perform global updating asynchronously among groups. Pang et al. [30] mitigates staleness by constraining malicious updates to lie within the upper-bound staleness and forcing compromised clients to upload their crafted models as quickly as the fastest clients.

However, these works do not explicitly handle non-IID data, which can amplify the negative effects of staleness and lead to gradient divergence [34]. In contrast, we propose a novel worker grouping algorithm that considers both the *edge heterogeneity* and the *data distribution* of workers, and assigns them to different groups accordingly. Therefore, our algorithm can reduce the communication overhead and improve the convergence of FL under non-IID data.

B. Federated Learning via Over-the-Air Computation

The first work that introduced over-the-air computation-based FL aggregation was by Zhu et al. [10], who leverage the broadband analog aggregation to achieve low-latency model aggregation. They further expand their work by using one-bit quantization at workers, followed by modulation and majority-voting-based decoding at the parameter server, to reduce the communication overhead [15]. Yang et al. [16] focus on the trade-off between communication and learning, and propose a method that maximizes the number of devices while minimizing the mean squared error (MSE) of gradient error. Another challenge in this approach is the bandwidth consumption. Amiri and Gündüz [17], [19] exploit the sparsity of the model update vector and project it into a low-dimensional space using random matrices. Their methods significantly reduce the bandwidth requirement, while preserving the accuracy of the model aggregation. Power control is another important factor that affects the performance of FL via AirComp. Zhang and Tao [35] formulate the power control problem as an optimization problem that minimizes the MSE of gradients, subject to average power constraints at each worker. Similarly, Cao et al. [18] conduct an analysis of the convergence of over-the-air computation FL under various power control policies to optimize transmit power. Yao et al. [20] jointly optimizes transmit power and digital combiner to minimize the MSE of gradient aggregation under power constraints, ensuring fairness among multiple FL tasks. Recently, researchers have made efforts to apply integrated sensing and communication (ISAC) empower edge AI, Wen et al. [21] investigates the ISAC for AirComp-based FL to exploit both benefits and further enhance resource utilization efficiency.

However, a common limitation in the aforementioned works is the requirement for all workers to concurrently transfer their local models for over-the-air aggregation, leading to the critical straggler problem. Although a few recent studies [31], [32] have proposed time-triggered schemes to integrate asynchronous federated learning with over-the-air computation, they rely solely on the synchronization point for aggregation, ignoring non-IID data among workers. To this end, our proposed mechanism groups workers taking into account their data distributions, allowing asynchronous FL to accelerate model training through over-the-air aggregation, while relaxing the synchronization requirements of this aggregation technology.

III. System Model

A. Federated Learning (FL)

For ease of expression, some key notations in this paper are listed in Table II. We perform federated learning over a set of workers $\mathcal{V} = \{v_1, v_2, \ldots, v_N\}$, with $|\mathcal{V}| = N > 1$. Each worker v_i trains a model on its local dataset d_i, with the size of $d_i \triangleq |d_i|$. Then the loss function of worker v_i is defined as

$$f_i(\mathbf{w}) \triangleq \frac{1}{d_i} \sum_{\xi \in d_i} f_i(\mathbf{w}; \xi), \tag{1}$$

TABLE II
KEY NOTATIONS

Symbol	Semantics
$\mathcal{V}$	The set of workers $\{v_1, v_2, ..., v_N\}$
$\mathcal{V}_j$	The set of workers in the j-th group
$\mathbf{V}$	The set of groups $\{\mathcal{V}_1, \mathcal{V}_2, ..., \mathcal{V}_M\}$
d_i, D_j, D	The data size of worker v_i/group $\mathcal{V}_j$/total
α_i	The proportion of the data size of worker v_i to the data size of its group
F, f_i	The global/local loss function
$\hat{\mathbf{w}}_t$	The error-free global model at round t
$\mathbf{w}_t$	The estimate of global model at round t
$\mathbf{g}_t^i$	The local gradient of worker v_i at round t
$\mathbf{y}_t$	The received signal on PS at round t
$\mathbf{z}_t$	The white Gaussian noise at round t
τ_t	The staleness at round t
σ_t^i	The scaling factor of worker v_i at round t
p_t^i	The transmit power of v_i at round t
E_t^i	The energy consumption of v_i at round t
κ_t	The denoising factor at round t

where $\mathbf{w}$ is the parameter vector, and $f_i(\mathbf{w}; \xi)$ is the loss over a sample ξ in dataset d_i.

The global dataset over all workers is $\mathcal{D}$, with size $D = |\mathcal{D}| = \sum_{v_i \in \mathcal{V}} d_i$. The global loss function on all the distributed datasets is defined as

$$F(\mathbf{w}) \triangleq \sum_{v_i \in \mathcal{V}} \frac{d_i}{D} f_i(\mathbf{w}). \tag{2}$$

The learning problem is to find the optimal parameter vector $\mathbf{w}^*$ so as to minimize $F(\mathbf{w})$, *i.e.*, $\mathbf{w}^* = \text{argmin}_{\mathbf{w}} F(\mathbf{w})$.

B. Grouping-Based Asynchronous Federated Learning via Over-the-Air Computation (Air-FedGA)

We propose the grouping-based asynchronous FL mechanism via AirComp, which is formally described in Alg. 1.

1) Worker Grouping: Workers in $\mathcal{V}$ are organized into M groups $\mathcal{V}_1, \ldots, \mathcal{V}_M$, satisfying $\bigcup_{j=1}^{M} \mathcal{V}_j = \mathcal{V}$ and $\mathcal{V}_j \bigcap \mathcal{V}_{j'} = \varnothing, \forall j \neq j'$. Let D_j denote the sum of the data size of workers in group $\mathcal{V}_j$, *i.e.*, $D_j = \sum_{v_i \in \mathcal{V}_j} d_i$. Let $\alpha_i = d_i / D_j$ denote the proportion of worker v_i's data size to its group data size.

2) Local Training: Let $\mathcal{V}_{j_t}$ denote the group that participating in the global updating at round t. If worker $v_i \notin \mathcal{V}_{j_t}$, it will not receive global model from the PS at round t, and its local gradient at round t is equal to that at round $t-1$, *i.e.*, $\mathbf{g}_t^i = \mathbf{g}_{t-1}^i$ (Line 9). On the contrary, if worker $v_i \in \mathcal{V}_{j_t}$, it receives the global model $\mathbf{w}_t$ and calculates its local gradient as

$$\mathbf{g}_t^i = \nabla f_i(\mathbf{w}_t, \xi_t^i), \tag{3}$$

where ξ_t^i is a sample uniformly chosen from the local dataset d_i at round t. Let τ_t be the interval between the current round t and the last received global model version by worker in group $\mathcal{V}_{j_t}$, called the *staleness*. Thus, $\mathbf{g}_t^i$ is equal to $\mathbf{g}_{t-\tau_t}^i$, and $\mathbf{g}_{t-\tau_t}^i$ is calculated from a previous version of the global model on v_i, *i.e.*,

$$\mathbf{g}_t^i = \mathbf{g}_{t-\tau_t}^i = \nabla f_i(\mathbf{w}_{t-\tau_t}, \xi_t^i). \tag{4}$$

Then, v_i sends a READY message to the parameter server (Lines 5-7).

3) Intra-Group Alignment: The parameter server maintains a set of variables $r_j, \forall j \in [1, M]$. Once the parameter server receives a READY message from a worker in group $\mathcal{V}_j$, r_j increases by 1 (Lines 14-24). If the parameter server has received all READY messages of worker in group $\mathcal{V}_j$, *i.e.*, $r_j = |\mathcal{V}_j|$, it sends the EXECUTE messages to workers in $\mathcal{V}_j$ and reset r_j as 0 (Lines 18-20).

4) Grouping Asynchronous Aggregation: On receiving the EXECUTE message, worker v_i transmits $\mathbf{g}_t^i$ and performs over-the-air aggregation simultaneously with all the other participating workers (Lines 10-11). Let h_t^i denote the wireless channel gain between worker v_i and the parameter server at round t, which is assumed to remain unchanged within one communication round. Then the transmit power of v_i at round t is set as

$$p_t^i = \frac{\alpha_i \sigma_t^i}{h_t^i}, \tag{5}$$

where σ_t^i is the power scaling factor of v_i at round t. According to the existing work [36], the transmission energy consumption

Algorithm 1 Grouping-based Asynchronous Federated Learning via Over-the-Air Computation (Air-FedGA)

1: **for** $j \in [1, M]$ **do**
2: $r_j = 0$
3: **for** $t = 1$ to T **do**
4: **Processing at Each Worker** v_i
5: **if** receive $\mathbf{w}_t$ from the PS **then**
6: Calculate local gradient $\mathbf{g}_t^i$ by Eq. (3)
7: Send READY message to the PS
8: **else**
9: $\mathbf{g}_t^i = \mathbf{g}_{t-1}^i$
10: **if** Receive EXECUTE message from the PS **then**
11: Transmit $\mathbf{g}_t^i$ simultaneously with all the other workers in group $\mathcal{V}_{j_t}$
12: **Processing at the Parameter Server**
13: **while True do**
14: **if** Receive READY message from worker v_i **then**
15: $j = \arg\{j \in [1, M] | v_i \in \mathcal{V}_j\}$
16: $r_j = r_j + 1$
17: **if** $r_j = |\mathcal{V}_j|$ **then**
18: $j_t = j$
19: $r_{j_t} = 0$
20: Send EXECUTE message to each $v_i \in \mathcal{V}_{j_t}$
21: Receive signal $\mathbf{y}_t$ by over-the-air aggregation
22: Update global model $\mathbf{w}_{t+1}$ according to Eq. (9)
23: Distribute $\mathbf{w}_t$ to each worker $v_i \in \mathcal{V}_{j_t}$
24: **break**
25: **return** global model $\mathbf{w}_T$

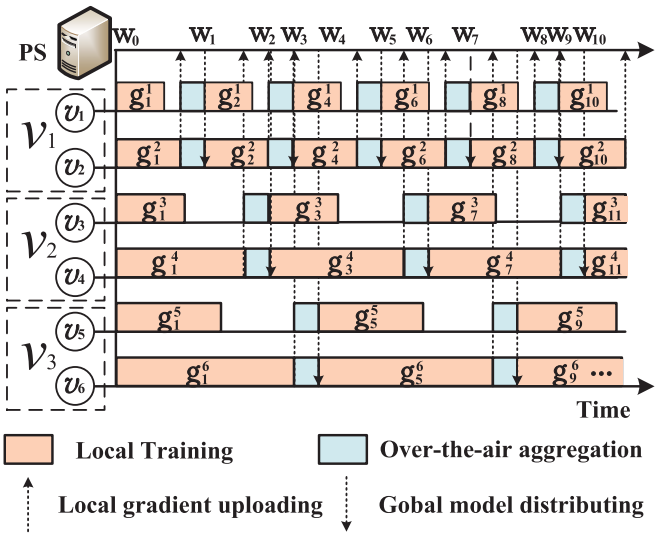


Fig. 2. The workflow of Air-FedGA.

on worker v_i at round t is given by

$$E_t^i = \|p_t^i \mathbf{g}_t^i\|_2^2. \tag{6}$$

As a result, their local models are aggregated over-the-air for the parameter server. If the aggregation is error-free, the global model can be obtained by

$$\hat{\mathbf{w}}_{t+1} = \mathbf{w}_t - \eta \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i. \tag{7}$$

where η is the learning rate (*i.e.*, step size). However, due to channel fading and noise, the received signal at the parameter server is given by

$$\mathbf{y}_t = \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} p_t^i h_t^i \mathbf{g}_t^i + \mathbf{z}_t = \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \sigma_t^i \mathbf{g}_t^i + \mathbf{z}_t, \tag{8}$$

where $\mathbf{z}_t$ is an additive white Gaussian noise (AWGN) vector with zero mean and variance σ_0^2. Therefore, the parameter server estimates the global model as

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta \mathbf{y}_t}{\sqrt{\kappa_t}}, \tag{9}$$

where κ_t is the denoising factor at round t. After aggregation, the parameter server distributes the global model $\mathbf{w}_t$ to all workers in $\mathcal{V}_{j_t}$.

In this paper, we consider AirComp in an ideal scenario with perfect channel state information (CSI), so that workers' signals overlap exactly with each other at the PS. Modeling non-ideal channel effects [37] is primarily a physical-layer work that is orthogonal to the mechanism proposed in this paper. We therefore plan to adopt such modeling to a follow-up study in which physical-layer effects are the main focus.

To visualize the procedure of Air-FedGA, we give an example in Fig. 2. There are 6 workers v_1-v_6 in the FL system, divided into 3 groups, *i.e.*, $\mathcal{V}_1 = \{v_1, v_2\}$, $\mathcal{V}_2 = \{v_3, v_4\}$, $\mathcal{V}_3 = \{v_5, v_6\}$ and $\mathbf{V} = \{\mathcal{V}_1, \mathcal{V}_2, \mathcal{V}_3\}$. For instance, workers v_1 andv_2 performs over-the-air aggregation simultaneouslyat round 1, *i.e.*, $\mathbf{w}_1 = \mathbf{w}_0 - \eta \frac{\alpha_1 \sigma_1^1 \mathbf{g}_1^1 + \alpha_2 \sigma_1^2 \mathbf{g}_1^2 + \mathbf{z}_1}{\sqrt{\kappa_1}}$. Since workers v_1 and v_2 receive the global model $\mathbf{w}_0$ at round 1, the staleness $\tau_1 = 0$. For another instance, workers v_5 and v_6 performs over-the-air aggregation simultaneously at round 4, *i.e.*, $\mathbf{w}_4 = \mathbf{w}_3 - \eta \frac{\alpha_5 \sigma_4^5 \mathbf{g}_4^5 + \alpha_6 \sigma_4^6 \mathbf{g}_4^6 + \mathbf{z}_4}{\sqrt{\kappa_4}}$. Since the last time workers v_5 and v_6 receive the global model $\mathbf{w}_0$ at round 1, $\mathbf{g}_4^5 = \mathbf{g}_1^5$ and $\mathbf{g}_4^6 = \mathbf{g}_1^6$, and the staleness $\tau_4 = 4 - 1 = 3$.

IV. Convergence Analysis

In this section, we first introduce several commonly adopted assumptions (Section IV-A). Based on these, we then present a rigorous convergence analysis of Air-FedGA (Section IV-B). Finally, derive insightful corollaries from the convergence bound and analyze the impact of key factors on training performance (Section IV-C).

A. Assumptions

We make the following commonly adopted assumptions [15], [20], [21] in FL setting.

Assumption 1: [Smoothness] The loss function F is λ-smooth with $\lambda > 0$, *i.e.*, for $\forall \mathbf{w}_1, \mathbf{w}_2$, $F(\mathbf{w}_2) - F(\mathbf{w}_1) \leq \langle \nabla F(\mathbf{w}_1), \mathbf{w}_2 - \mathbf{w}_1 \rangle + \frac{\lambda}{2} \|\mathbf{w}_2 - \mathbf{w}_1\|^2$.

Assumption 2: [Gradient Bound] The expected squared norm of stochastic gradients is uniformly bounded, *i.e.*, $\forall i, t$, $\mathbb{E}\|\mathbf{g}_t^i\|^2 \leq G^2$.

B. Convergence Analysis for Air-FedGA

For asynchronous aggregation among groups, we denote ψ_j as the relative frequency of group $\mathcal{V}_j$ participating in the global aggregation, satisfying $\sum_{\mathcal{V}_j \in \mathbf{V}} \psi_j = 1$. To facilitate analysis, we define the loss function of group $\mathcal{V}_j$ as

$$F_j(\mathbf{w}) \triangleq \sum\nolimits_{v_i \in \mathcal{V}_j} \frac{d_i}{D_j} f_i(\mathbf{w}) = \sum\nolimits_{v_i \in \mathcal{V}_j} \alpha_i f_i(\mathbf{w}). \tag{10}$$

We also define $\Gamma_j \triangleq \max_{\mathbf{w}} \|\nabla F(\mathbf{w}) - \nabla F_j(\mathbf{w})\|$ as the upper bound of gradient divergence between the group $\mathcal{V}_j$'s and the global loss functions [38]. From Eqs. (7)-(9), the model aggregation error of caused by the over-the-air aggregation at round t is given by

$$\varepsilon_t = \frac{\mathbf{w}_{t+1} - \hat{\mathbf{w}}_{t+1}}{\eta} = \frac{\mathbf{y}_t}{\sqrt{\kappa_t}} - \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i. \tag{11}$$

Then the mean squared error (MSE) [18] at round t is calculated as

$$\begin{aligned} \mathbb{E}\|\varepsilon_t\|^2 &= \mathbb{E}\left\| \frac{\mathbf{y}_t}{\sqrt{\kappa_t}} - \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i \right\|^2 \\ &= \mathbb{E}\left\| \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i \left(\frac{\sigma_t^i}{\sqrt{\kappa_t}} - 1 \right) + \frac{\mathbf{z}_t}{\sqrt{\kappa_t}} \right\|^2 \\ &\leq 2 \left[G^2 \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \left(\frac{\sigma_t^i}{\sqrt{\kappa_t}} - 1 \right)^2 + \frac{\sigma_0^2}{\kappa_t} \right]. \end{aligned} \tag{12}$$

Theorem 1: Suppose that the Assumption 1 and 2 is hold and let $\frac{1}{2\lambda} < \eta < \frac{1}{\lambda}$. $\mathbf{w}_0$ is the initial global model. After the model updating Eq. (9) is performed T times, the trained global model $\mathbf{w}_T$ satisfies

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2 \leq \underbrace{\frac{C_1}{T}}_{\text{vanishing}} + \underbrace{C_2 \tau_{\max}^2}_{\text{staleness}} + \underbrace{C_3 \sum\nolimits_{\mathcal{V}_j \in \mathbf{V}} \psi_j \Gamma_j^2}_{\text{non-IID term}} + \underbrace{\frac{C_4}{T} \sum_{t=0}^{T-1} \mathbb{E}\|\varepsilon_t\|^2}_{\text{MSE term}}.$$

where $C_1 = \frac{2\lambda(F(\mathbf{w}_0)-F(\mathbf{w}^*))}{2\lambda\eta-1}$, $C_2 = \frac{4\lambda^3\eta^3G^2}{2\lambda\eta-1}$, $C_3 = \frac{4\lambda\eta M}{2\lambda\eta-1}$ and $C_4 = \frac{2\lambda^2\eta^2}{2\lambda\eta-1}$.

Proof: From Eq. (11), we deduce that

$$\mathbf{w}_{t+1} - \mathbf{w}_t = -\frac{\eta \mathbf{y}_t}{\sqrt{\kappa_t}} = -\eta \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i - \eta\varepsilon_t. \tag{13}$$

Combining Assumption 1 and Eq. (13), we have

$$\begin{aligned}
&F(\mathbf{w}_{t+1}) \\
&\le F(\mathbf{w}_t) + \langle \nabla F(\mathbf{w}_t), \mathbf{w}_{t+1} - \mathbf{w}_t \rangle + \frac{\lambda}{2}\|\mathbf{w}_{t+1} - \mathbf{w}_t\|^2 \\
&\le F(\mathbf{w}_t) - \eta \langle \nabla F(\mathbf{w}_t), \frac{\mathbf{y}_t}{\sqrt{\kappa_t}} \rangle + \frac{\lambda\eta^2}{2}\|\frac{\mathbf{y}_t}{\sqrt{\kappa_t}}\|^2 \\
&\le F(\mathbf{w}_t) - \eta \langle \nabla F(\mathbf{w}_t), \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i \rangle - \eta \langle \nabla F(\mathbf{w}_t), \varepsilon_t \rangle \\
&\quad + \frac{\lambda\eta^2}{2}\|\sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i\|^2 + \frac{\lambda\eta^2\|\varepsilon_t\|^2}{2} \\
&\quad + \lambda\eta^2 \langle \varepsilon_t, \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i \rangle.
\end{aligned} \tag{14}$$

From Eqs. (4) and (10), we deduce that

$$\begin{aligned}
\mathbb{E}\left[\sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \mathbf{g}_t^i\right] &= \mathbb{E}\left[\sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \nabla f_i(\mathbf{w}_{t-\tau_t}, \xi_t^i)\right] \\
&= \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \nabla f_i(\mathbf{w}_{t-\tau_t}) \\
&= \nabla F_{j_t}(\mathbf{w}_{t-\tau_t}).
\end{aligned} \tag{15}$$

Take the expectation on both sides of Eq. (14), we have

$$\begin{aligned}
\mathbb{E}[F(\mathbf{w}_{t+1})] \le \mathbb{E}[F(\mathbf{w}_t)] &- \eta\mathbb{E}\langle \nabla F(\mathbf{w}_t), \nabla F_{j_t}(\mathbf{w}_{t-\tau_t}) \rangle \\
&+ \frac{\lambda\eta^2}{2}\mathbb{E}\|\nabla F_{j_t}(\mathbf{w}_{t-\tau_t})\|^2 + \frac{\lambda\eta^2\mathbb{E}\|\varepsilon_t\|^2}{2} \\
&+ \mathbb{E}\langle \eta\varepsilon_t, -\nabla F(\mathbf{w}_t) + \lambda\eta\nabla F_{j_t}(\mathbf{w}_{t-\tau_t}) \rangle.
\end{aligned} \tag{16}$$

By using the AM-GM Inequality for the last term of Eq. (16), we have

$$\begin{aligned}
&\langle \eta\varepsilon_t, -\nabla F(\mathbf{w}_t) + \lambda\eta\nabla F_{j_t}(\mathbf{w}_{t-\tau_t}) \rangle \\
&\le \frac{\lambda\eta^2\|\varepsilon_t\|^2}{2} + \frac{\|\nabla F(\mathbf{w}_t) - \lambda\eta\nabla F_{j_t}(\mathbf{w}_{t-\tau_t})\|^2}{2\lambda} \\
&= \frac{\lambda\eta^2\|\varepsilon_t\|^2}{2} + \frac{\|\nabla F(\mathbf{w}_t)\|^2}{2\lambda} + \frac{\lambda\eta^2\|\nabla F_{j_t}(\mathbf{w}_{t-\tau_t})\|^2}{2} \\
&\quad - \eta\langle \nabla F(\mathbf{w}_t), \nabla F_{j_t}(\mathbf{w}_{t-\tau_t}) \rangle.
\end{aligned} \tag{17}$$

Since $\eta < \frac{1}{\lambda}$, by taking Eq. (17) into Eq. (16), we deduce that

$$\begin{aligned}
&\mathbb{E}[F(\mathbf{w}_{t+1})] \\
&\le \mathbb{E}[F(\mathbf{w}_t)] - 2\eta\mathbb{E}\langle \nabla F(\mathbf{w}_t), \nabla F_{j_t}(\mathbf{w}_{t-\tau_t}) \rangle \\
&\quad + \lambda\eta^2\mathbb{E}\|\nabla F_{j_t}(\mathbf{w}_{t-\tau_t})\|^2 + \lambda\eta^2\mathbb{E}\|\varepsilon_t\|^2 + \frac{\mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2}{2\lambda} \\
&\le \mathbb{E}[F(\mathbf{w}_t)] + \eta\mathbb{E}\|\nabla F(\mathbf{w}_t) - \nabla F_{j_t}(\mathbf{w}_{t-\tau_t})\|^2 \\
&\quad - \eta\mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2 + \lambda\eta^2\mathbb{E}\|\varepsilon_t\|^2 + \frac{\mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2}{2\lambda}.
\end{aligned} \tag{18}$$

We derive the second term of Eq. (18) as

$$\begin{aligned}
&\mathbb{E}\|\nabla F(\mathbf{w}_t) - \nabla F_{j_t}(\mathbf{w}_{t-\tau_t})\|^2 \\
&= \mathbb{E}\|\nabla F(\mathbf{w}_t) - \nabla F(\mathbf{w}_{t-\tau_t}) + \nabla F(\mathbf{w}_{t-\tau_t}) - \nabla F_{j_t}(\mathbf{w}_{t-\tau_t})\|^2 \\
&\le 2\mathbb{E}\|\nabla F(\mathbf{w}_t) - \nabla F(\mathbf{w}_{t-\tau_t})\|^2 \\
&\quad + 2\mathbb{E}\|\nabla F(\mathbf{w}_{t-\tau_t}) - \nabla F_{j_t}(\mathbf{w}_{t-\tau_t})\|^2 \\
&\le 2\lambda^2\mathbb{E}\|\mathbf{w}_t - \mathbf{w}_{t-\tau_t}\|^2 + 2\Gamma_{j_t}^2
\end{aligned} \tag{19}$$

By using Jensen's inequality, we further derive that

$$\begin{aligned}
\mathbb{E}\|\mathbf{w}_t - \mathbf{w}_{t-\tau_t}\|^2 &= \mathbb{E}\|\sum\nolimits_{r=t-\tau_t}^{t-1}(\mathbf{w}_{r+1} - \mathbf{w}_r)\|^2 \\
&= \mathbb{E}\|\sum\nolimits_{r=t-\tau_t}^{t-1}\eta\sum\nolimits_{v_i \in \mathcal{V}_{j_r}}\alpha_i\mathbf{g}_r^i\|^2 \\
&\le \eta^2\tau_t\sum\nolimits_{r=t-\tau_t}^{t-1}\mathbb{E}\|\sum\nolimits_{v_i \in \mathcal{V}_{j_r}}\alpha_i\mathbf{g}_r^i\|^2 \\
&\le \eta^2\tau_{\max}^2 G^2
\end{aligned} \tag{20}$$

Taking Eq. (19) into Eq. (18), we have

$$\begin{aligned}
\mathbb{E}[F(\mathbf{w}_{t+1})] \le \mathbb{E}[F(\mathbf{w}_t)] &+ 2\lambda^2\eta^3\tau_{\max}^2 G^2 + 2\eta\Gamma_{j_t}^2 \\
&- \left(\eta - \frac{1}{2\lambda}\right)\mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2 + \lambda\eta^2\mathbb{E}\|\varepsilon_t\|^2.
\end{aligned} \tag{21}$$

Since $\eta > \frac{1}{2\lambda}$, we rearrange terms and further derive that

$$\begin{aligned}
\mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2 \le &\frac{2\lambda(\mathbb{E}[F(\mathbf{w}_t) - F(\mathbf{w}_{t+1})])}{2\lambda\eta - 1} \\
&+ \frac{4\lambda^3\eta^3\tau_{\max}^2 G^2 + 4\lambda\eta\Gamma_{j_t}^2 + 2\lambda^2\eta^2\mathbb{E}\|\varepsilon_t\|^2}{2\lambda\eta - 1}.
\end{aligned} \tag{22}$$

Summing Eq. (22) from $t = 0$ to $T-1$, and noting that $\mathbb{E}[F(\mathbf{w}_T)] \ge F(\mathbf{w}^*)$, we have

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F(\mathbf{w}_t)\|^2 \\
&\le \frac{2\lambda(F(\mathbf{w}_0) - F(\mathbf{w}^*))}{T(2\lambda\eta - 1)} + \frac{4\lambda^3\eta^3\tau_{\max}^2 G^2}{2\lambda\eta - 1} \\
&\quad + \frac{4\lambda\eta M}{2\lambda\eta - 1}\sum_{\mathcal{V}_j \in \mathbf{V}}\psi_j\Gamma_j^2 + \frac{2\lambda^2\eta^2}{T(2\lambda\eta - 1)}\sum_{t=0}^{T-1}\mathbb{E}\|\varepsilon_t\|^2 \\
&= \frac{C_1}{T} + C_2\tau_{\max}^2 + C_3\sum_{\mathcal{V}_j \in \mathbf{V}}\psi_j\Gamma_j^2 + \frac{C_4}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\varepsilon_t\|^2,
\end{aligned} \tag{23}$$

where $C_1 = \frac{2\lambda(F(\mathbf{w}_0)-F(\mathbf{w}^*))}{2\lambda\eta-1}$, $C_2 = \frac{4\lambda^3\eta^3G^2}{2\lambda\eta-1}$, $C_3 = \frac{4\lambda\eta M}{2\lambda\eta-1}$ and $C_4 = \frac{2\lambda^2\eta^2}{2\lambda\eta-1}$.■

C. Discussions

We can draw some meaningful corollaries from Theorem 1.

Corollary 1: Air-FedGA convergent to a neighbourhood of a stationary point with rate $O\left(\frac{1}{T}\right)$, and the limiting neighborhood size is determined by the maximum staleness $O(\tau_{\max}^2)$, the degree of non-IID data $O\left(\sum_{\mathcal{V}_j \in \mathbf{V}}\psi_j\Gamma_j^2\right)$, and the average MSE $O\left(\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\|\varepsilon_t\|^2\right)$.

Corollary 2: The greater the degree of non-IID data among groups, the larger the value of Γ_j for each group $\mathcal{V}_j$, and the higher convergence bound. Given IID data among groups, then $\Gamma_j = 0$ for $\forall\mathcal{V}_j \in \mathbf{V}$, the convergence bound can be reduced.

Corollary 1 and 2 indicate that workers can be organized to minimize $\sum_{\mathcal{V}_j \in \mathbf{V}}\psi_j\Gamma_j^2$, which is equivalent to reducing the

degree of non-IID, thereby tightening the convergence bound and enhancing training performance.

Corollary 3: The convergence bound decreases as the upper bound of staleness $\tau_{\max}$ decreases. $\tau_{\max}$ depends partly on the number M of groups. For example, if $M = 1$, then $\tau_{\max} = 0$, and ρ takes the minimum value.

Corollary 3 shows that we can decrease the convergence factor ρ by decreasing the number M of groups. However, it does not mean a short convergence time, because the completion time of a single round depends on the worker with the maximum completion time within a group. Consequently, less groups will result in a longer completion time of a single round. Empirically, we set the number of groups as $|\mathbf{V}| = M = \lfloor\sqrt{N}\rfloor$ [39], since this balances straggler delay (which increases as M get smaller) against degree of non-IID data (which increases as M get larger), yielding a trade-off between edge heterogeneity and non-IID data. For example, with $N = 100$ workers we take $M = \sqrt{100} = 10$, so each group has roughly 10 workers. This limits the waiting to about 10 stragglers per aggregation while dividing approximately 10 distinct data partitions.

Corollary 4: The power scaling factor σ_t^i of each worker v_i and the denoising factor κ_t at round t are related to the MSE $\mathbb{E}\|\varepsilon_t\|^2$.

Corollary 4 shows that we can decouple the AirComp aggregation parameters σ_t^i and κ_t from the convergence bound to minimize MSE, which will be elaborated in Section V-B.

V. Problem Formulation and Algorithm Description

Instead of deploying a heuristic algorithm for worker grouping without performance guarantees in our previous work [1], we redesign a more principled worker grouping algorithm based on the convergence analysis in Section IV. Specifically, we first formulate a convergence bound minimization problem given time and energy constraints (Section V-A), and then design a iterative power control algorithm to determine the decoupled communication parameters (*i.e.*, power scaling factor, the denoising factor) (Section V-B). Next, we transform the original problem into a convex optimization problem using the quadratic transform and relaxation techniques (Section V-C). Finally, we design a rounding-based worker grouping algorithm to solve the problem and derive its approximate ratio (Section V-D).

A. Problem Formulation

To exploit AirComp for low-latency model aggregation [10], the model aggregation time is calculated as

$$L^u = \frac{q}{R} L^s, \tag{24}$$

where q is the dimension of the trained model, R is the number of sub-channels, and L^s is the symbol duration of an OFDM symbol.

Let $x_{i,j}$ denote the indicator for whether worker v_i belongs to group $\mathcal{V}_j$ or not. The grouping strategy in the whole system is denoted as $\boldsymbol{x} = \{x_{i,j}\}_{v_i\in\mathcal{V},\mathcal{V}_j\in\mathbf{V}}$. Let l_i denote the local training time on worker v_i, which is assumed to be estimated by the historical measurements. $\Delta l = \max_{v_i\in\mathcal{V}}\{l_i\} - \min_{v_i\in\mathcal{V}}\{l_i\}$ is the difference between the maximum and minimum local training time of workers in $\mathcal{V}$. The time for all workers in group $\mathcal{V}_j$ to complete local training is determined by the worker with the longest training time. Then the completion time for $\mathcal{V}_j$ to complete local training and model uploading via over-the-air aggregation is calculated as

$$L_j = \max_{v_i\in\mathcal{V}_j}\{l_i\} + L^u = \max_{v_i\in\mathcal{V}}\{x_{i,j} l_i\} + L^u. \tag{25}$$

Therefore, the number of updates that group $\mathcal{V}_j$ participates in per unit time is $\frac{1}{L_j}$. Since all groups participate in global updating asynchronously, the average completion time of one round is estimated as

$$\overline{L} \approx \frac{1}{\frac{1}{L_1} + \frac{1}{L_2} + \ldots + \frac{1}{L_M}} = \frac{1}{\sum_{\mathcal{V}_j\in\mathbf{V}} \frac{1}{L_j}}. \tag{26}$$

According to Theorem 1, we consider the convergence bound

$$\begin{aligned}&\mathcal{B}(T, \tau_{\max}, \{\Gamma_j^2\}, \{\mathbb{E}\|\varepsilon_t\|^2\})\\ &= \frac{C_1}{T} + C_2\tau_{\max}^2 + C_3\sum\nolimits_{\mathcal{V}_j\in\mathbf{V}} \psi_j\Gamma_j^2 + \frac{C_4}{T}\sum\nolimits_{t=0}^{T-1}\mathbb{E}\|\varepsilon_t\|^2.\end{aligned} \tag{27}$$

To speed up the convergence of Air-FedGA, we formulate the problem as follows:

$$\begin{aligned}\textbf{(P1)}: \min_{\boldsymbol{\sigma},\boldsymbol{\kappa},\boldsymbol{x}} \ & \mathcal{B}(T, \tau_{\max}, \{\Gamma_j^2\}, \{\mathbb{E}\|\varepsilon_t\|^2\})\\ \textbf{s.t.} \ & \overline{L}T \le \hat{L} && \text{(28a)}\\ & E_t^i \le \hat{E}^i, \quad \forall v_i\in\mathcal{V}, t\in[T] && \text{(28b)}\\ & L_j - L^u - l_i \le \varepsilon\Delta l, \quad \forall v_i\in\mathcal{V}_j, \mathcal{V}_j\in\mathbf{V} && \text{(28c)}\\ & \sum\nolimits_{\mathcal{V}_j\in\mathbf{V}} x_{i,j} = 1, \quad \forall v_i\in\mathcal{V} && \text{(28d)}\\ & x_{i,j}\in\{0,1\}, \quad v_i\in\mathcal{V}, \mathcal{V}_j\in\mathbf{V}. && \text{(28e)}\end{aligned}$$

The first inequality (28a) represents that the total training time cannot exceed the maximum threshold $\hat{L}$. The second set of inequalities (28b) represent that each worker v_i is subject to a maximum energy budget $\hat{E}^i$ at each round t. The third set of inequalities (28c) ensures the local training time of workers within each group is similar (*e.g.*, $\varepsilon = 0.3$). The fourth equality (28d) represent that each worker belongs to a unique group. Our target is to determine the power scaling factors $\boldsymbol{\sigma} = \{\sigma_t^i | v_i \in \mathcal{V}, t \in [T]\}$, the denoising factors $\boldsymbol{\kappa} = \{\kappa_t | t\in[T]\}$ and the grouping strategy $\boldsymbol{x}$ to minimize the convergence bound the global loss function after T rounds of training, *i.e.*, $\min_{\boldsymbol{\sigma},\boldsymbol{\kappa},\boldsymbol{x}} \mathcal{B}(T, \tau_{\max}, \{\Gamma_j^2\}, \{\mathbb{E}\|\varepsilon_t\|^2\})$.

B. Power Control

Since the power scaling factor σ_t^i of each worker v_i and the denoising factor κ_t at round t are related to $\mathbb{E}\|\varepsilon_t\|^2$ in the optimization objective in **P1**, we decouple the process of solving for $\boldsymbol{\sigma}_t = \{\sigma_t^i | \forall v_i \in \mathcal{V}_{j_t}\}$ and κ_t from **P1**. From Eq. (12), we let $H_t = 2G^2\sum_{v_i\in\mathcal{V}_{j_t}} \alpha_i \left(\frac{\sigma_t^i}{\sqrt{\kappa_t}} - 1\right)^2 + \frac{\sigma_0^2}{\kappa_t}$ as the upper bound of MSE $\mathbb{E}\|\varepsilon_t\|^2$, then we determine $\boldsymbol{\sigma}_t$ and κ_t to minimize H_t,

$$\begin{aligned}\textbf{(P2)}: \min_{\boldsymbol{\sigma}_t,\kappa_t} \ & H_t\\ \textbf{s.t.} \ & E_t^i \le \hat{E}^i, \qquad \forall v_i\in\mathcal{V}, t\in[T] && \text{(29a)}\end{aligned}$$

Algorithm 2 Iterative Algorithm for Power Control

Input: Initial scaling factor $\boldsymbol{\sigma}_t = \{\sigma_t^i | \forall v_i \in \mathcal{V}_{j_t}\}$ of round t

Output: convergent $\boldsymbol{\sigma}_t^*$ and κ_t^*

while $\frac{\|\boldsymbol{\sigma}_t^* - \boldsymbol{\sigma}_t\|}{\|\boldsymbol{\sigma}_t^*\|} > \epsilon$ or $\frac{|\kappa_t^* - \kappa_t|}{\kappa_t^*} > \epsilon$ **do**

  $\boldsymbol{\sigma}_t^* = \boldsymbol{\sigma}_t$, $\kappa_t^* = \kappa_t$

  $\kappa_t = \left[\frac{G^2 \sum_{v_i \in \mathcal{V}_{j_t}} \alpha_i (\sigma_t^i)^2 + \sigma_0^2}{G^2 \sum_{v_i \in \mathcal{V}_{j_t}} \alpha_i \sigma_t^i}\right]^2$

  for each $v_i \in \mathcal{V}_{j_t}$ **do**

    $\sigma_t^i = \min\left\{\sqrt{\kappa_t}, \frac{h_t^i \sqrt{\hat{E}^i}}{\alpha_i G}\right\}$

$\boldsymbol{\sigma}_t^* = \boldsymbol{\sigma}_t = \{\sigma_t^i | \forall v_i \in \mathcal{V}_{j_t}\}$, $\kappa_t^* = \kappa_t$

return convergent $\boldsymbol{\sigma}_t^*$ and κ_t^*

Note that $\boldsymbol{\sigma}_t$ and κ_t are coupled in **P2**. Therefore, we address **P2** by adopting the alternating optimization method [18], which is formally described in Alg. 2. The main idea is to alternately fix $\boldsymbol{\sigma}_t/\kappa_t$ and determine the value of $\kappa_t/\boldsymbol{\sigma}_t$ to optimize H_t. After several iterations, convergent $\boldsymbol{\sigma}_t^*$ and κ_t^* are obtained.

At each iteration, we first optimize the denoising factors κ_t under given scaling factors $\boldsymbol{\sigma}_t^*$. Let $\hat{\kappa}_t = \frac{1}{\sqrt{\kappa_t}}$, H_t is transformed to

$$H_t = 2G^2 \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i \left(\sigma_t^i \hat{\kappa}_t - 1\right)^2 + \sigma_0^2 \hat{\kappa}_t^2. \tag{30}$$

The partial derivative of H_t with respect to $\hat{\kappa}_t$ is calculated as

$$\frac{\partial H_t}{\partial \hat{\kappa}_t} = 4G^2 \sum\nolimits_{v_i \in \mathcal{V}_{j_t}} \alpha_i [(\sigma_t^i)^2 \hat{\kappa}_t - \sigma_t^i] + 2\sigma_0^2 \hat{\kappa}_t. \tag{31}$$

Since H_t is convex with respect to $\hat{\kappa}_t$, the necessary condition for minimization is given by setting the partial derivative to zero, *i.e.*, $\frac{\partial H_t}{\partial \hat{\kappa}_t} = 0$. Solving this equation yields the optimal $\hat{\kappa}_t = \frac{G^2 \sum_{v_i \in \mathcal{V}_{j_t}} \alpha_i \sigma_t^i}{G^2 \sum_{v_i \in \mathcal{V}_{j_t}} \alpha_i (\sigma_t^i)^2 + \sigma_0^2}$, *i.e.*,

$$\kappa_t = \left[\frac{G^2 \sum_{v_i \in \mathcal{V}_{j_t}} \alpha_i (\sigma_t^i)^2 + \sigma_0^2}{G^2 \sum_{v_i \in \mathcal{V}_{j_t}} \alpha_i \sigma_t^i}\right]^2. \tag{32}$$

Next, we optimize the scaling factor σ_t^i under given denoising factor κ_t. On the one hand, the partial derivative of H_t with respect to σ_t^i is calculated as

$$\frac{\partial H_t}{\partial \sigma_t^i} = 4G^2 \alpha_i \left(\frac{\sigma_t^i}{\kappa_t} - \frac{1}{\sqrt{\kappa_t}}\right). \tag{33}$$

Given that H_t is convex with respect to σ_t^i, we similarly set $\frac{\partial H_t}{\partial \sigma_t^i} = 0$. Solving it shows that H_t is minimized when $\sigma_t^i = \sqrt{\kappa_t}$, provided that this value lies within the feasible region. On the other hand, from Assumption 2 and constraints Eq. (28b), for $\forall v_i \in \mathcal{V}$, we have the following inequality:

$$E_t^i = \|p_t^i \mathbf{g}_t^i\|_2^2 \leq \left(\frac{\alpha_i \sigma_t^i}{h_t^i}\right)^2 G^2 \leq \hat{E}^i. \tag{34}$$

This leads to the bound $\sigma_t^i \leq \frac{h_t^i \sqrt{\hat{E}^i}}{\alpha_i G}$ for $\forall v_i \in \mathcal{V}$. Therefore, if κ_t is given, H_t can be minimized when

$$\sigma_t^i = \min\left\{\sqrt{\kappa_t}, \frac{h_t^i \sqrt{\hat{E}^i}}{\alpha_i G}\right\}. \tag{35}$$

At last, with a given threshold ϵ, the convergent $\boldsymbol{\sigma}_t^*$ and κ_t^* can be obtained by alternate optimization for iterations.

C. Reformulate the Worker Grouping Problem

After decoupling $\boldsymbol{\sigma} = \{\boldsymbol{\sigma}_t, t \in [T]\}$ and $\boldsymbol{\kappa} = \{\kappa_t | t \in [T]\}$ from problem **P1**, we obtain the following optimization problem:

$$\textbf{(P3)}: \min_{\boldsymbol{x}} \mathcal{B}(T, \tau_{\max}, \{\Gamma_j^2\})$$
$$\textbf{s.t.} \text{ (28a), (28c), (28d) and (28e).} \tag{36a}$$

Next, we transfer the three critical term T, $\tau_{\max}$ and $\{\Gamma_j^2\}_{\mathcal{V}_j \in \mathbf{V}}$ of the objective in **P3**.

1) T: From constraint Eq. (28a), the number of training rounds satisfy
$$T \leq \frac{\hat{L}}{\overline{L}}. \tag{37}$$
It is obvious that when we set $T = \lfloor \frac{\hat{L}}{\overline{L}} \rfloor$, $\mathcal{B}$ reaches the minimum. To simplify the analysis, we assume that $\hat{L}$ is an integer multiple of $\overline{L}$, *i.e.*, $T = \frac{\hat{L}}{\overline{L}}$.

2) $\tau_{\max}$: We observe that the group with the largest staleness factor $\tau_{\max}$ is also the group with the longest completion time. Therefore, $\tau_{\max}$ can be estimated as
$$\tau_{\max} = \max_{\mathcal{V}_j \in \mathbf{V}} \{L_j\} \cdot \sum_{\mathcal{V}_j \in \mathbf{V}} \frac{1}{L_j} = L_{\max} \sum_{\mathcal{V}_j \in \mathbf{V}} \frac{1}{L_j} = \frac{L_{\max}}{\overline{L}}, \tag{38}$$
where $L_{\max} = \max_{\mathcal{V}_j \in \mathbf{V}} \{L_j\} = \max_{v_i \in \mathcal{V}} \{l_i\}$ is a constant, which represent the maximum completion time among all workers.

3) $\{\Gamma_j^2\}_{\mathcal{V}_j \in \mathbf{V}}$: Similar to Eq. (26), the relative participation frequency ψ_j of group $\mathcal{V}_j$ is estimated by
$$\psi_j = \frac{\frac{1}{L_j}}{\sum_{\mathcal{V}_j \in \mathbf{V}} \frac{1}{L_j}} = \frac{\overline{L}}{L_j}. \tag{39}$$
Thus, we re-express the following term as
$$\sum\nolimits_{\mathcal{V}_j \in \mathbf{V}} \psi_j \Gamma_j^2 = \overline{L} \cdot \sum\nolimits_{\mathcal{V}_j \in \mathbf{V}} \frac{\Gamma_j^2}{L_j}. \tag{40}$$

Therefore, the convergence bound is transferred as

$$\mathcal{B}(\overline{L}, \{\Gamma_j^2\}) = \frac{C_1 + C_4 \sum_{t=0}^{T-1} H_t}{\hat{L}} \overline{L} + C_2 L_{\max}^2 \frac{1}{\overline{L}^2} + C_3 \overline{L} \cdot \sum\nolimits_{\mathcal{V}_j \in \mathbf{V}} \frac{\Gamma_j^2}{L_j}, \tag{41}$$

which is a function that first decreases and then increases with respect to $\overline{L}$. However, the coefficients C_1-C_4 depend on model- and loss-specific properties (*e.g.*, λ and $F(\mathbf{w}^*)$) that are generally difficult to quantify. As a result, it is challenging to obtain the exact $\overline{L}$ that minimizes $\mathcal{B}$. To resolve this, we introduce a tunable hyperparameter $\widetilde{L}$ and constrain $\overline{L}$ in the optimization problem by $\overline{L} \leq \widetilde{L}$. The value of $\widetilde{L}$ is selected empirically through experiments. $\{\Gamma_j^2\}_{\mathcal{V}_j \in \mathbf{V}}$ captures the effect of non-IID data among the groups. A higher degree of non-IID data leads to larger values of Γ_j for each group $\mathcal{V}_j$. However, the terms $\sum_{\mathcal{V}_j \in \mathbf{V}} \frac{\Gamma_j^2}{L_j}$ and $\overline{L}$ are interdependent

and cannot be minimized simultaneously. To reconcile these conflicts, we reformulate the worker grouping problem as a constrained optimization problem prioritizing $\sum_{\mathcal{V}_j\in\mathbf{V}}\frac{\Gamma_j^2}{L_j}$ while bounding $\overline{L}$:

$$\textbf{(P4)}: \min_{\boldsymbol{x}} \sum_{\mathcal{V}_j\in\mathbf{V}}\frac{\Gamma_j^2}{L_j}$$
$$\text{s.t. } \overline{L}\le\widetilde{L} \tag{42a}$$
$$\text{(28c), (28d) and (28e).} \tag{42b}$$

D. Rounding-Based Algorithm for Worker Grouping

1) Transform to Convex Optimization Problem: **P4** is difficult to solve due to combinatorial group assignments and non-convex terms, we first transform and relax it into a convex optimization problem.

1) **Linearizing Non-smooth Variable:** Constraint Eq. (42a) involves the variable $L_j=\max_{v_i\in\mathcal{V}}\{x_{i,j}l_i\}+L^u$ which is non-smooth due to the max operator. To linearize it, we replace the equality with the inequality
$$x_{i,j}l_i+L^u\le L_j. \tag{43}$$
2) **Estimating Gradient Divergences:** In federated learning, the PS cannot inspect each worker's local data directly. However, it is observe that there is an implicit connection between the data distribution on a worker and the gradient weights trained on that worker [33]. Therefore, by collecting gradients after one additional round of local training, the PS can infer similarity between workers and group workers accordingly. Similar to the existing works [33], [38], [40], the PS distributes an initial model $\mathbf{w}_{\text{init}}$ to all workers. Each worker v_i computes its local gradient $\mathbf{g}_i=\nabla f_i(\mathbf{w}_{\text{init}})$, and returns $\mathbf{g}_i$ to the PS. The PS estimates gradient divergence Γ_j for each group $\mathcal{V}_j$ as
$$\begin{aligned}\hat{\Gamma}_j&=\|\nabla F(\mathbf{w}_{\text{init}})-\nabla F_j(\mathbf{w}_{\text{init}})\|\\&=\|\sum\nolimits_{v_i\in\mathcal{V}}\alpha_i\mathbf{g}_i-\sum\nolimits_{v_i\in\mathcal{V}}x_{i,j}\alpha_i\mathbf{g}_i\|\end{aligned} \tag{44}$$
3) **Relaxing Discrete Variables:** $x_{i,j}$ is binary in constraint Eq. (42b). To render the problem convex, we relax this to a continuous variable, *i.e.*, $x_{i,j}\in[0,1]$ for $\forall v_i\in\mathcal{V},\mathcal{V}_j\in\mathbf{V}$.
4) **Quadratic Transform for Multiple-Ratio Objective:** The optimizationobjective in **P4** is given by $\sum_{\mathcal{V}_j\in\mathbf{V}}\frac{\hat{\Gamma}_j^2}{L_j}$, which is a multiple-ratio fractional programming problem and thus non-convex. To address the non-convexity, we employ the quadratic transform technique [41] (an extended version of Dinkelbach's transform [42]), which decouples each ratio $\frac{\hat{\Gamma}_j^2}{L_j}$ by introducing auxiliary variables ϕ_j: $\frac{\hat{\Gamma}_j^2}{L_j}=\min_{\phi_j}\{2\phi_j\hat{\Gamma}_j-\phi_j^2L_j\}$, with the optimal value attained at $\phi_j=\frac{\hat{\Gamma}_j}{L_j}$ (derived by setting the derivative of the quadratic expression *w.r.t.* ϕ_j to zero). This transforms the objective into a convex sum of quadratic terms.

Taking the above into account, we transform **P5** into the following convex optimization problem:

$$\textbf{(P5)}: \min_{\boldsymbol{x},\boldsymbol{\phi}} \sum_{\mathcal{V}_j\in\mathbf{V}}(2\phi_j\hat{\Gamma}_j-\phi_j^2L_j)$$
$$\text{s.t. } x_{i,j}l_i+L^u\le L_j,\quad\forall v_i\in\mathcal{V},\mathcal{V}_j\in\mathbf{V} \tag{45a}$$
$$\sum\nolimits_{\mathcal{V}_j\in\mathbf{V}}\frac{1}{L_j}\ge\frac{1}{\widetilde{L}} \tag{45b}$$
$$L_j-L^u-l_i\le\varepsilon\Delta l,\quad\forall v_i\in\mathcal{V}_j,\mathcal{V}_j\in\mathbf{V} \tag{45c}$$
$$\sum\nolimits_{\mathcal{V}_j\in\mathbf{V}}x_{i,j}=1,\quad\forall v_i\in\mathcal{V} \tag{45d}$$
$$x_{i,j}\in[0,1],\quad\forall v_i\in\mathcal{V},\mathcal{V}_j\in\mathbf{V} \tag{45e}$$

Algorithm 3 Worker Grouping Algorithm for Air-FedGA

Input: Estimated gradient divergences $\Gamma_j,\forall\mathcal{V}_j\in\mathbf{V}$, local training times $l_i,\forall v_i\in\mathcal{V}$
Output: Final grouping strategy $\boldsymbol{x}$
Step 1: Transform to a convex optimization problem
Construct **P5** by relaxation and quadratic transform
Step 2: Obtain fraction solution Iteratively
Initial a grouping strategy $\boldsymbol{x}$
while $\frac{\|\boldsymbol{x}^*-\boldsymbol{x}\|}{\|\boldsymbol{x}^*\|}>\epsilon$ **do**
  $\boldsymbol{x}^*=\boldsymbol{x}$
  Update $\boldsymbol{\phi}=\{\phi_j|\mathcal{V}_j\in\mathbf{V}\}$ by Eq. (46)
  Update $\boldsymbol{x}$ by solving the convex optimization problem **P5** under fixed $\boldsymbol{\phi}$.
Step 3: Rounding to 0-1 Solution
for each $v_i\in\mathcal{V}$ **do**
  Find v_{j^*} with the largest $\widetilde{x}_{i,j^*}$ among $\widetilde{x}_{i,j},\forall\mathcal{V}_j\in\mathbf{V}$
  for $\mathcal{V}_j\in\mathbf{V}$ **do**
    Obtain integer solution $\widehat{x}_{i,j}$ by Eq. (47)
return final grouping strategy $\boldsymbol{x}$

2) Obtain Fraction Solution Iteratively: We then optimize the decision variable $\boldsymbol{x}$ and the auxiliary variable $\boldsymbol{\phi}=\{\phi_j|\mathcal{V}_j\in\mathbf{V}\}$ using an alternating optimization approach. As detailed in Alg. 3, when $\boldsymbol{x}$ is held fixed, the optimal value of each $\boldsymbol{\phi}$ can be obtained in closed form as
$$\phi_j=\frac{\hat{\Gamma}_j}{L_j}=\frac{\|\sum_{v_i\in\mathcal{V}}\alpha_i\mathbf{g}_i-\sum_{v_i\in\mathcal{V}}x_{i,j}\alpha_i\mathbf{g}_i\|}{\max_{v_i\in\mathcal{V}}\{x_{i,j}l_i\}+L^u},\forall\mathcal{V}_j\in\mathbf{V} \tag{46}$$
Conversely, when $\boldsymbol{\phi}$ is held fixed, **P5** is a convex problem, and can be solved using standard convex optimization tools, *e.g.*, CVXPY package [43]. This iterative process is repeated until convergence, *i.e.*, until the difference between successive iterations falls below a predetermined threshold.

3) Rounding-Based Solution: In this section, we determine how to determine the assignment of each worker v_i to groups. Let the optimal fractional solution derived from **P5** be $\widetilde{x}_{i,j}$. For each worker v_i, we obtain an integer solution about its assignment $\widehat{x}_{i,j}$ by rounding the maximum fractions among $\widetilde{x}_{i,j},\forall\mathcal{V}_j\in\mathbf{V}$. Formally, let $\mathcal{V}_{j^*}$ be the group for worker v_i such that $\widetilde{x}_{i,j^*}=\max_{\mathcal{V}_j\in\mathbf{V}}\widetilde{x}_{i,j}$, we perform the following rounding operations:
$$\widehat{x}_{i,j}=\begin{cases}1 & \text{if } j=j^*\\0 & \text{otherwise.}\end{cases} \tag{47}$$

Remark 1: In Alg. 3, each worker's local training time l_i is treated as fixed. However, in practice l_i may vary over time because of changing system load. To make the grouping algorithm robust to such dynamics, we periodically re-grouping workers using recent estimates of each worker's completion time, which expands the grouping algorithm to system load variation scenario with small additional overhead. The corresponding experimental results are provide in Section VI-B.

VI. Performance Evaluation

A. System Setup

We use PyTorch to simulate a large-scale federated learning system, which consists of one parameter server and 100 workers. Each worker simulates an individual machine and trains a local model on its own dataset. We conduct our experiments on a deep learning workstation with a 10-core Intel Xeon CPU (Silver 4210R) and 4 NVIDIA GeForce RTX 3090 GPUs with 24GB GDDR6X. The system environment is Ubuntu 22.04, CUDA v11.7, and cuDNN v8.5.0.

1) Models and Datasets: We adopt four publicly available, well-established benchmark datasets to conduct extensive experiments, enabling a comprehensive evaluation of our method across varying levels of complexity and scale.

- **MNIST** [44] comprises a collection of handwritten digits (from '0' to '9'), includes 60,000 training samples and 10,000 testing samples. We use MNIST as a lightweight baseline for evaluating algorithms on low-capacity tasks.
- **CIFAR-10** [45] composed of 60,000 low-resolution natural images, which is divided into 10 classes, each containing 6,000 images. The dataset is further split into 50,000 training images and 10,000 test images.
- **ImageNet-100** is a subset of dataset ImageNet [46], which contains 1,281,167 training images, 50,000 validation images and 100,000 test images, spread across 1,000 categories. To accommodate the limited resources of edge clients, we construct a subset of ImageNet by randomly selecting the samples of 100 out of 1000 categories.
- **Digit-5** [47] is a multi-domain digit classification dataset composed of five digit datasets: MNIST [44], SVHN [48], USPS [49], SynthDigits [50], and MNIST-M [50]. These source datasets come from different visual domains and thus present distinct feature distributions while sharing the same label space.

To implement the non-IID data among workers, we adopt both label skew and feature skew methods to partition the original datasets. For the first three dataset, we adopt the label skewed method [51]. Specifically, the data in MNIST labeled as '0' are distributed to workers v_1-v_{10}, labeled as '1' are distributed to workers v_{11}-v_{20},..., and labeled as '9' are distributed to workers v_{91}-v_{100}. For Digit-5 dataset, we use the feature skew method to create a feature skew non-IID setup. Specifically, the MNIST dataset is assigned to workers v_1-v_{20}, the USPS dataset to workers v_{21}-v_{40}, the SVHN dataset to workers v_{41}-v_{60}, the SynthDigits dataset to workers v_{61}-v_{80}, and the MNIST-M dataset to workers v_{81}-v_{100}.

Three different models with distinct structures are implemented on the aforementioned datasets:

- **LR [52] on MNIST.** The logistic regression (LR in short), which is constructed of a fully connected network with two hidden layers with 512 units, is adopted for the MNIST dataset.
- **CNN [53] on MNIST and CIFAR-10.** The plain CNN models are tailored for the MNIST and CIFAR-10. 1) For MNIST: It consists of two 5×5 convolution layers (20, 50 channels), two fully-connected layers with 800 and 500 units, and a softmax layer with 10 units. 2) For CIFAR-10: The CNN consists of two 5×5 convolution layers (32, 64 channels), two fully-connected layers with 1600 and 512 units, and a softmax layer with 10 units.
- **VGG-16 [54] on ImageNet-100.** The VGG-16 model, which consists of 13 convolutional layers with a kernel of 3×3, followed by two dense layers and a softmax output layer, is adopted for the ImageNet-100 dataset.
- **ResNet-18 [55] on Digit-5.** The ResNet-18 model, which consists of 16 convolutional layers organized into residual blocks, followed by a global average pooling layer and a final fully-connected softmax output layer, is adopted for the Digit-5 dataset.

2) Simulation of Edge Heterogeneity: Let $\hat{l}_i$ denote the actual local training time on v_i. Due to the limitations of training resources and the large-scale scenario, we actually deploy the virtual workers v_1-v_{100} on the single workstation for experimentation, their local training times are roughly equal, *i.e.*, $\hat{l}_1 \approx \hat{l}_2 \approx \ldots \approx \hat{l}_{100}$. To simulate the edge heterogeneity, we introduce a scaling factor c_i for each worker v_i, which is a random float number drawn uniformly from [1,10]. Then, we set the local training time on worker v_i as $l_i = c_i\hat{l}_i$, which means that after completing local training, v_i waits for 0-9 times before sending the READY message to the parameter server. This adjusted time l_i is then used to calculate its training completion time and recorded in a dynamically maintained list, $\mathbf{L}$. By monitoring the training completion times of all workers recorded in $\mathbf{L}$, we determine when each group performs over-the-air aggregations. Additionally, we set the bandwidth $B = 1$MHz, the noisy variance $\sigma_0^2 = 1$W, and the energy constraints $\hat{E}^i = 10$J for each worker in each round.

3) Benchmarks and Metrics: To highlight the benefits of applying AirComp to asynchronous federated learning, we evaluate our **Air-FedGA** mechanism against two OMA-based mechanisms and three AirComp-based mechanisms.

- **FedAvg:** [11] A classic OMA-based synchronous mechanism, where all workers participating in each round of global aggregation.
- **TiFL:** [29] An OMA-based group asynchronous mechanism, which organize workers into groups according to their communication time with the parameter server, and perform global updating asynchronously among groups.
- **Air-FedAvg:** [18] The version of using AirComp technique to implement the FedAvg mechanism with optimal power control.
- **Dynamic:** [36] An AirComp-based synchronous mechanism, which dynamically selects a subset of workers for each round of global aggregation, while the rest remain idle.

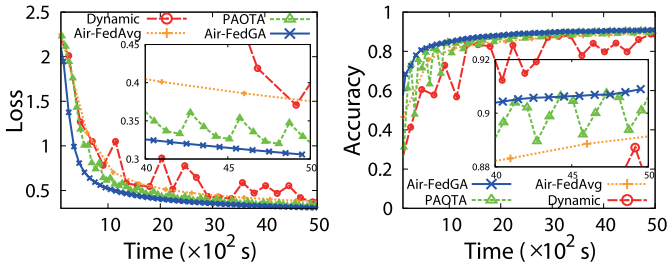


Fig. 3. Loss/Accuracy vs. Time (LR on MNIST). *Left*: Loss; *Right*: Accuracy.

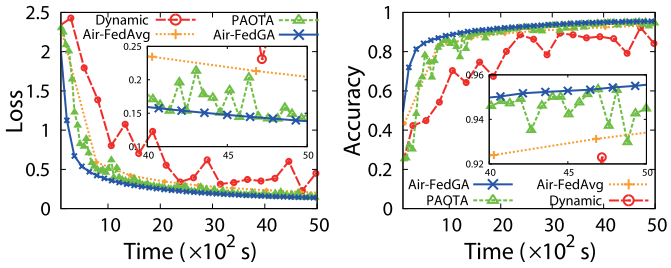


Fig. 4. Loss/Accuracy vs. Time (CNN on MNIST). *Left*: Loss; *Right*: Accuracy.

- **PAOTA:** [32] An AirComp-based time-triggered asynchronous mechanism. Each worker that completes local training pauses until a predefined synchronization point, at which all available workers simultaneously perform over-the-air aggregation.

To evaluate the training performance, we adopt the following performance metrics. 1) *Loss Function* reflects the training process of the model and whether convergence has been achieved. 2) *Accuracy* is the most common performance metric in classification problems, which is defined as the proportion of right data classified by the model to all test data. 3) *Training Time* is adopted to measure the training rate.

B. Evaluation Results

In this section, we first compare the performance of our proposed Air-FedGA with other benchmarks in terms of loss function and accuracy. We then demonstrate the advantages of Air-FedGA in scalability, handling heterogeneity, and handling non-IID data.

1) Loss Function and Accuracy: Figs. 3–7 illustrate the loss and accuracy curves over time for LR trained on MNIST, CNN trained on MNIST, CNN trained on CIFAR-10, VGG-16 trained on ImageNet-100, and ResNet-18 trained on Digit-5 respectively. As a control experiment, in this set of experimental results, our Air-FedGA is compared with three AirComp-based mechanisms, *i.e.*, Air-FedAvg, Dynamic and PAOTA.

Across all four models and datasets, Air-FedGA outperforms Air-FedAvg, Dynamic and PAOTA in terms of convergence speed and accuracy. For example, in the LR-MNIST experiment shown in Fig. 3, Air-FedGA achieves a stable accuracy of 90.7% after 5000s of training, while Air-FedAvg, Dynamic and PAOTA only reach 88.9%, 86.1% and 89.2%, respectively. Moreover, Air-FedGA attains a stable 80% accuracy in 482s, which can accelerate the model training

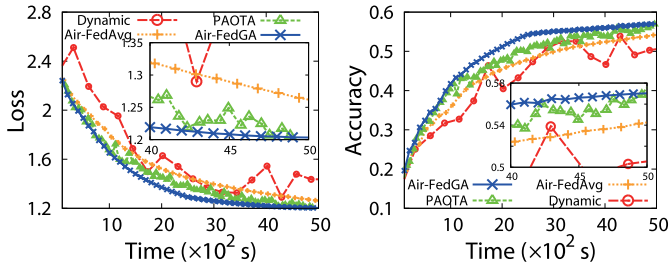


Fig. 5. Loss/Accuracy vs. Time (CNN on CIFAR-10). *Left*: Loss; *Right*: Accuracy.

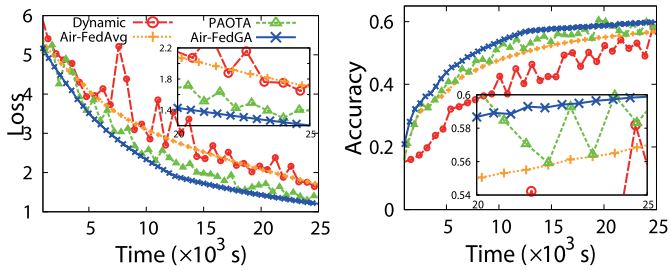


Fig. 6. Loss/Accuracy vs. Time (VGG-16 on ImageNet-100). *Left*: Loss; *Right*: Accuracy.

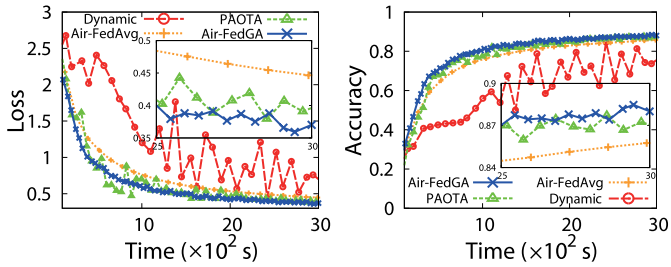


Fig. 7. Loss/Accuracy vs. Time (ResNet-18 on Digit-5). *Left*: Loss; *Right*: Accuracy.

by 2.22×, 9.29× and 2.28× compared with Air-FedAvg (1072s), Dynamic (4482s) and PAOTA (1102s), respectively. For CNN on MNIST shown in Fig. 4, Air-FedGA reaches 95.7% accuracy after 5000s, surpassing Air-FedAvg (93.7%), Dynamic (83.9%) and PAOTA (94.1%). Air-FedGA reaches stable 80% accuracy in 398s, which can accelerate the model training by 1.83×, 11.33× and 3.14× compared with Air-FedAvg (730s), Dynamic (4510s) and PAOTA (1252s), respectively. On the more complex CIFAR-10 dataset shown in Fig. 5, Air-FedGA reaches 57.5% accuracy after 5000s, while Air-FedAvg, Dynamic and PAOTA reach 54.2%, 50.3% and 55.1%, respectively. For VGG-16 on ImageNet-100 shown in Fig. 6, Air-FedGA achieves 59.6% accuracy after 25000s, exceeding Air-FedAvg (57.1%), Dynamic (48.2%) and PAOTA(58.2%). For the feature distribution skew setting, we train ResNet-18 on Digit-5, as shown in Fig. 7. After 3000s of training, Air-FedGA achieves a stable accuracy of 87.9%, outperforming Air-FedAvg (85.6%), Dynamic (75.2%), and PAOTA (86.9%). The reason for the superior performance of Air-FedGA is that it adopts a group asynchronous updating mechanism that reduces the waiting time of workers, while Air-FedAvg suffers from long waiting time due to its rigid synchronization across all workers.

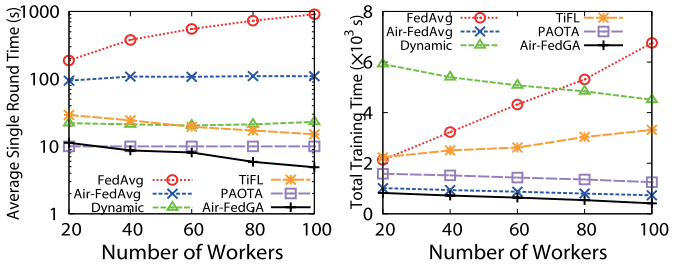


Fig. 8. Training Time vs. Number of Workers. *Left*: Single Round; *Right*: Total.

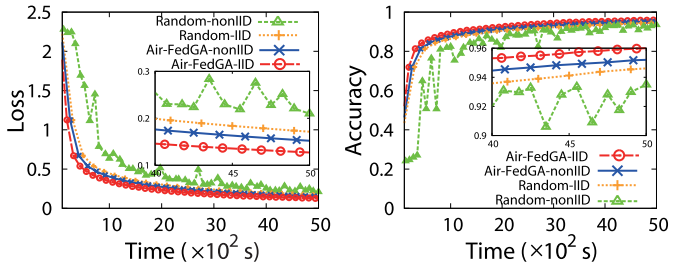


Fig. 9. Effect of the worker grouping algorithm (CNN on MNIST). *Left*: Loss; *Right*: Accuracy.

We also observe that the loss and accuracy curves of Dynamic jitter significantly across training rounds. This volatility arises from Dynamic's round-by-round worker selection, which introduces bias into the global model updating [51]. Moreover, since only a subset of workers is selected in each round while the others remain idle, the overall training efficiency is reduced. In addition, the loss and accuracy curves of PAOTA have a slight jitter. This is because PAOTA sets a fixed synchronization point for the model aggregation of waited workers, while ignoring non-IID data among workers. In contrast, Air-FedGA employs a grouping-based asynchronous updating mechanism, where workers are assigned into groups considering their local data distributions, and each group performs global updating asynchronously. This design not only minimizes idle time but also decrease the degree of non-IID data (label or feature skew) at the group level, substantially reducing the fluctuation in loss and accuracy compared to Dynamic and PAOTA. As a result, Air-FedGA achieves higher training efficiency and better handles non-IID data.

2) AirComp for Scalability: To evaluate the scalability of our proposed Air-FedAvg, we compare it against both OMA-based mechanisms (FedAvg and TiFL) and AirComp-based mechanisms (Air-FedAvg, Dynamic and PAOTA). Fig. 8 evaluate the average single round time (left) and the total training time required to reach 80% accuracy when training a CNN on MNIST, as the number of workers N increases. Note that since the training time varies widely in magnitude, the left plot is plotted using logarithmic scales.

As shown in the left plot, FedAvg's single round time increases linearly with N, as its OMA-based synchronous aggregation requires each worker to upload its model to the server. In contrast, both Air-FedAvg and Dynamic maintain a nearly constant single round time independent of N by leveraging over-the-air aggregation. PAOTA is set with fixed synchronization points [32], so its single round time is also constant. In addition, mechanisms that adopt grouping asynchronous updating—TiFL and Air-FedGA—demonstrate a reduction in single round time as N increases. This is because larger values of N result in more asynchronous groups, which facilitate parallel updates, decrease worker idle times, and allow communication to overlap with computation. Notably, Air-FedGA achieves even shorter single round time compared to TiFL by further reducing the communication latency via its over-the-air aggregation technique.

The right plot of Fig. 8 shows the total training time of different mechanisms to achieve stable 80% accuracy. It is observed that the total training time of the methods without over-the-air aggregation, *i.e.*, FedAvg and TiFL, increases with the increase of N. In contrast, for the mechanisms employing over-the-air aggregation, *i.e.*, Air-FedAvg, Dynamic, PAOTA and Air-FedGA, the total training time decreases with an increasing N. In particular, by combining grouping asynchronous updating (which reduces idle time) with over-the-air aggregation (which shortens communication latency), Air-FedGA achieves the most dramatic reduction in total training time among all mechanisms. For example, when $N = 100$, the total training time of FedAvg, Dynamic, TiFL, PAOTA, Air-FedAvg and Air-FedGA is 6755s, 4510s, 3319s, 1252s, 730s and 413s, respectively.

In summary, our Air-FedGA, by combining asynchronous grouping updating with over-the-air aggregation, achieves the best scalability among all mechanisms.

3) Effect of the Worker Grouping Algorithm: Fig. 9 illustrates the effect of our worker grouping algorithm on the loss and accuracy of CNN training on MNIST under both IID and non-IID settings. For comparison, we also consider a baseline algorithm that randomly organizes 100 workers into 10 groups. As shown, our grouping algorithm under non-IID data achieves performance comparable to that under IID data. For example, after 5000s of training, the proposed algorithm achieves accuracies of 95.2% and 96.1% for non-IID and IID data, respectively. This improvement arises because our grouping strategy aims to minimize the degree of non-IID data among groups, thereby enhancing convergence performance. In contrast, the random grouping algorithm exhibits significant fluctuations in both loss and accuracy under non-IID data, due to the higher degree of non-IID data among groups. Moreover, our grouping algorithm under non-IID data even outperforms the random grouping algorithm under IID data. This is because our method also considers edge heterogeneity, grouping workers with similar completion times together. In comparison, the random grouping strategy may assign workers with highly imbalanced completion times to the same group, thereby slowing down the training process.

4) Effect of the Power Control Algorithm: Fig. 10 illustrates the effect of our power control algorithm on the loss and accuracy of CNN training on MNIST. For comparison, we evaluate the performance of Air-FedGA against its variant Air-FedGA w/o PC, which excludes the proposed power control algorithm. In the latter, each worker sets its scaling factor to achieve perfect signal magnitude alignment under constraint (28b) [56]. As shown in the figure, after 5000s of training,

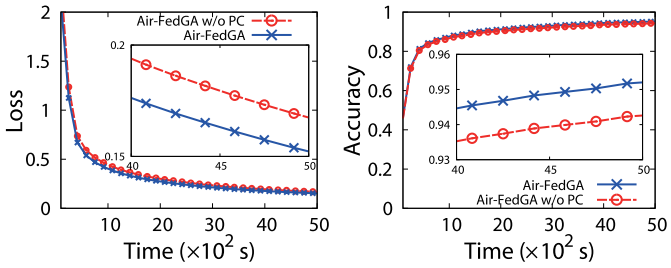


Fig. 10. Effect of the power control algorithm (CNN on MNIST). *Left*: Loss; *Right*: Accuracy.

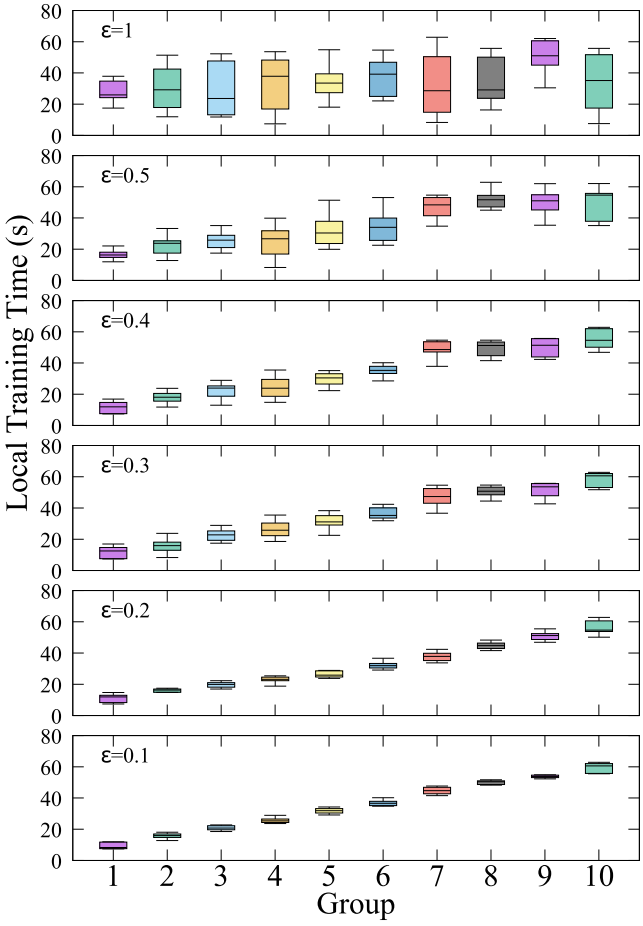


Fig. 11. Grouping of workers with different local training time when $\xi = 0.3$.

Air-FedGA and Air-FedGA w/o PC achieve accuracies of 95.2% and 94.2%, respectively. The performance degradation of Air-FedGA w/o PC arises because the aligned signal magnitude is dominated by the device with the weakest channel conditions. This effect significantly weakens the aggregate signal strength and equivalently amplifies the noise power.

5) Handling Edge Heterogeneity: To address edge heterogeneity, we intuitively tend to group workers with similar local training time together. Recall that the parameter ε in constraint (28c) controls the allowed variation in local training times within each group. Fig. 11 presents box plots of local training

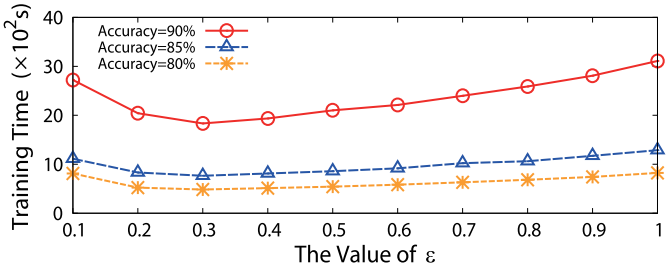


Fig. 12. Training time under different values of ε.

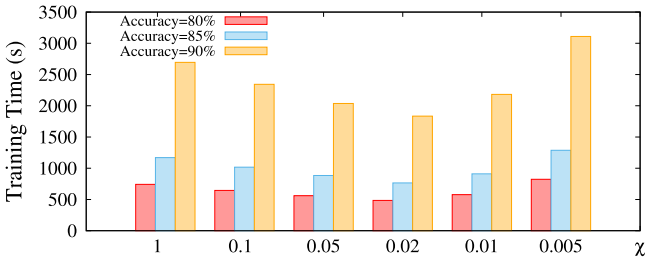


Fig. 13. Training time under different values of $\chi = \widetilde{L}/l_{\max}$.

times for 100 workers under varying ε values from 0.1 to 1. As shown in this figure, when ε is large, the local training times of workers within each group vary widely. For example, when $\varepsilon = 1$, for all groups, the minimum training time among workers in the group is below 18s while the maximum exceeds 38s. As ε decreases, workers with similar local training time are generally assigned within the same group. For example, when $\varepsilon = 0.1$, workers in Group 1 have local training times between 7.4s and 11.9s, whereas those in Group 10 range from 55.5s to 62.8s. This reflects the algorithm's adaptation to tighter constraints by grouping workers with similar local training times together, thereby addressing edge heterogeneity.

To determine the best choice of ε, Fig. 12 illustrates the training time required for CNN to achieve accuracy of 80%, 85%, and 90% on the MNIST dataset, with ε ranging from 0.1 to 1. The results indicate that the shortest training times occur at $\varepsilon = 0.3$, reaching 80%, 85%, and 90% accuracy in 485s, 765s, and 1834s, respectively. As ε increases toward 1, the training time gradually increases. This is because the training duration of each group—determined by the slowest worker—increases with larger values of ε, resulting in an increase in the total training time. For example, when $\varepsilon = 1$, the time required to reach 80%, 85% and 90% accuracy is 823s, 1288s and 3110s, respectively. Conversely, when ε becomes too small (*e.g.*, 0.1), required training time also increases (812s, 1280s, and 2723s for 80%, 85%, and 90% accuracy, respectively). This is because overly strict grouping based solely on training time can exacerbate the degree of non-IID data among groups, negatively affecting convergence. Therefore, $\varepsilon = 0.3$ achieves a favorable balance between handing edge heterogeneity and non-IID data.

6) Influence of \widetildeL: Recall that $\widetilde{L}$ in (42a) is a hyper-parameter that constrains the average round completion time $\widetilde{L}$. To determine the most appropriate value of $\widetilde{L}$, Fig. 13 illustrates the training time required for CNN to achieve

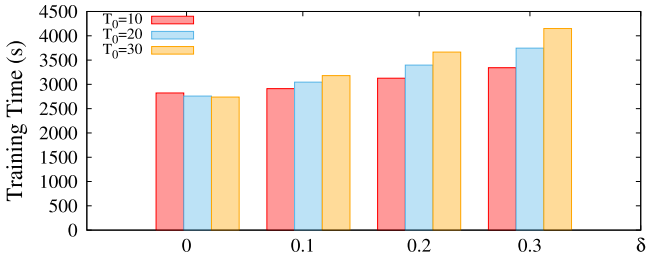


Fig. 14. Training time under system load variation.

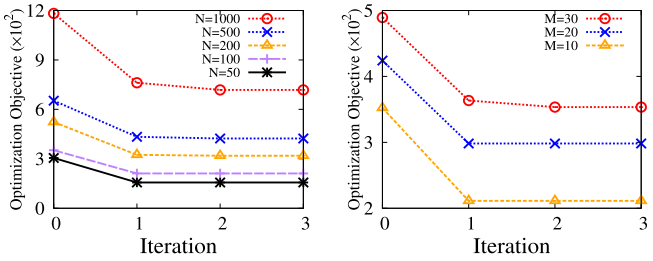


Fig. 15. Worker grouping algorithm iterations. *Left*: Different N under $M = 10$; *Right*: Different M under $N = 100$.

accuracy of 80%, 85%, and 90% on the MNIST dataset under varying values of χ. The results show that training time is prolonged when χ is set too high or too low. For example, when $\chi = 1$, the training times to reach 80%, 85%, and 90% accuracy are 742s, 1170s, and 2695s, respectively, while with $\chi = 0.005$, these times increase to 823s, 1288s and 3110s. In contrast, the training time is minimized when $\chi = 0.02$, requiring only 485s, 765s and 1834s to reach the same accuracy levels. Therefore, we set $\widetilde{L} = 0.02\, l_{\max}$ for all the other experiments.

7) Handling System Load Dynamic: To evaluate the our grouping algorithm under dynamic system load, we make each worker's local training time $l_i^{(t)}$ of worker v_i time-varying across rounds. At each round t, $l_i^{(t)}$ is drawn uniformly from the interval $[l_i^{(t-1)}(1-\delta), l_i^{(t-1)}(1+\delta)]$, where δ controls the volatility. Every T_0 rounds the PS re-runs the grouping algorithm using the latest l_i values to adapt groups to the current load status. Fig. 14 shows the time required for training CNN on MNIST to reach 90% accuracy as δ varies from 0 to 0.3 for three re-grouping intervals $T_0 = 10$, 20, 30. We observe that training time increases with δ. For example, when $T_0 = 30$, the total training time is 2738s, 3180s, 3665s and 4150s when the value of δ is 0, 0.1, 0.2 and 0.3, respectively. This occurs because higher volatility in local training times may cause the existing grouping strategy to become suboptimal. However, more frequent re-grouping (smaller T_0) can mitigate this effect and reduce the overall training time by timely updating the grouping strategy. For example, when $\delta = 0.3$, the training time is 4150s, 3746s and 3342s when T_0 is chosen as 30, 20 and 10, respectively. The experimental results indicate that the re-grouping method can effectively reduce the negative impact of system load variation.

8) The Running Time of the Worker Grouping Algorithm: Fig. 15 illustrates the evolution of the optimization objective

TABLE III
NUMBER OF ITERATIONS REQUIRED FOR CONVERGENCE

Iterations	1	2	3	4
Number of Cases	63	31	6	0

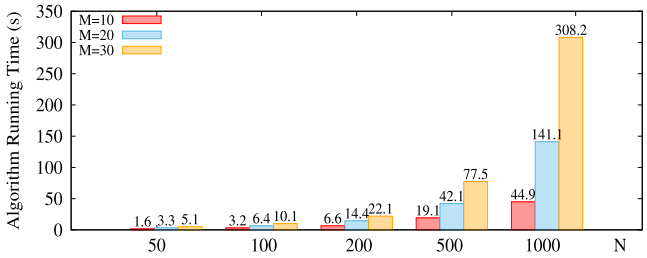


Fig. 16. The running time of the worker grouping algorithm.

during the iterative process of our worker grouping algorithm. As shown, the algorithm converges within three iterations for different numbers of workers N (ranging from 50 to 1000) and groups M (ranging from 10 to 30). We further run the grouping algorithm 100 times, and the corresponding number of iterations required for convergence is summarized in Table III. It is observed that the algorithm can converge with only a few iterations (*e.g.*, fewer than 3 in all cases).

Fig. 16 shows the running time of the worker grouping algorithm under different number N of workers from 50 to 1000, and number M of groups from 10 to 30. As shown, the running time increases superlinearly as either N or M grows. For example, when M is fixed at 10, the running time increases from 1.6s to 44.9s as N increases from 50 to 1000. Whereas, when N is fixed at 1000, the running time increases from 44.9s to 308.2s as M increases from 10 to 30. Nevertheless, even in large-scale settings with 1000 workers, the running time of the grouping algorithm remains negligible compared to the overall model training time. In extremely large-scale environments, the convex optimization process may become time-consuming. In such cases, distributed optimization techniques [57] can be employed to partition the workers into smaller subsets and solve corresponding convex subproblems in parallel. This extension lies beyond the scope of this paper and will not be further discussed here.

VII. CONCLUSION

In this paper, we have designed Air-FedGA, a mechanism that overcomes the limitations of deploying asynchronous federated learning via over-the-air computation, to address the communication constraints and edge heterogeneity in large-scale FL scenarios. We have analyzed the convergence of Air-FedGA, and explored the quantitative relationship between the convergence bound and key factors, *e.g.*, the maximum staleness, the degree of non-IID data among groups, and the AirComp aggregation MSE. Our analysis provides clear reasoning on how our grouping and power control strategy jointly improve communication efficiency and convergence stability. Specifically, reducing the degree of non-IID data among groups and AirComp aggregation MSE directly tightens the

convergence bound, leading to faster and more stable training for Air-FedGA. Extensive simulations have demonstrated that the proposed solutions significantly accelerate FL compared with the state-of-the-art solutions. For future work, we will explore hierarchical and device-to-device architectures to scale AirComp-based FL across larger-scale networks that involves more edge devices.

REFERENCES

[1] Q. Ma et al., "Air-FedGA: A grouping asynchronous federated learning mechanism exploiting over-the-air computation," 2025, *arXiv:2507.05704*.

[2] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Towards an intelligent edge: Wireless communication meets machine learning," 2018, *arXiv:1809.00343*.

[3] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, "In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning," *IEEE Netw.*, vol. 33, no. 5, pp. 156–165, Sep. 2019.

[4] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," 2016, *arXiv:1610.02527*.

[5] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, Apr. 2019, pp. 1387–1395.

[6] X. Mo and J. Xu, "Energy-efficient federated edge learning with joint communication and computation design," *J. Commun. Inf. Netw.*, vol. 6, no. 2, pp. 110–124, Jun. 2021.

[7] B. Luo, X. Li, S. Wang, J. Huang, and L. Tassiulas, "Cost-effective federated learning in mobile edge networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3606–3621, Dec. 2021.

[8] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Convergence time optimization for federated learning over wireless networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 4, pp. 2457–2471, Apr. 2021.

[9] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient federated learning over wireless communication networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 1935–1949, Mar. 2021.

[10] G. Zhu, Y. Wang, and K. Huang, "Broadband analog aggregation for low-latency federated edge learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 1, pp. 491–506, Jan. 2020.

[11] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. AISTATS*, 2016, pp. 1273–1282.

[12] X. Fan, Y. Wang, Y. Huo, and Z. Tian, "Joint optimization of communications and federated learning over the air," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 4434–4449, Jun. 2022.

[13] B. Nazer and M. Gastpar, "Computation over multiple-access channels," *IEEE Trans. Inf. Theory*, vol. 53, no. 10, pp. 3498–3516, Oct. 2007.

[14] X. Cao, G. Zhu, J. Xu, Z. Wang, and S. Cui, "Optimized power control design for over-the-air federated edge learning," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 342–358, Jan. 2022.

[15] G. Zhu, Y. Du, D. Gündüz, and K. Huang, "One-bit over-the-air aggregation for communication-efficient federated edge learning: Design and convergence analysis," *IEEE Trans. Wireless Commun.*, vol. 20, no. 3, pp. 2120–2135, Mar. 2021.

[16] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, Mar. 2020.

[17] M. M. Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent over-the-air," *IEEE Trans. Signal Process.*, vol. 68, pp. 2155–2169, 2020.

[18] X. Cao, G. Zhu, J. Xu, and S. Cui, "Transmission power control for over-the-air federated averaging at network edge," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 5, pp. 1571–1586, May 2022.

[19] M. M. Amiri and D. Gündüz, "Federated learning over wireless fading channels," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3546–3557, May 2020.

[20] J. Yao, W. Xu, G. Zhu, Z. Yang, K. Huang, and D. Niyato, "Over-the-air multitask federated learning via cost-effective hybrid beamforming," *IEEE Trans. Signal Process.*, vol. 73, pp. 3977–3993, 2025.

[21] D. Wen et al., "Integrated sensing, communication, and computation for over-the-air federated edge learning," *IEEE Trans. Wireless Commun.*, early access, Aug. 28, 2025, doi: 10.1109/TWC.2025.3598997.

[22] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," 2019, *arXiv:1903.03934*.

[23] W. Wu, L. He, W. Lin, R. Mao, C. Maple, and S. Jarvis, "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead," *IEEE Trans. Comput.*, vol. 70, no. 5, pp. 655–668, May 2021.

[24] Z. Chen, W. Yi, H. Shin, and A. Nallanathan, "Adaptive semi-asynchronous federated learning over wireless networks," *IEEE Trans. Commun.*, vol. 73, no. 1, pp. 394–409, Jan. 2025.

[25] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-IID data," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2020, pp. 15–24.

[26] S. Zheng et al., "Asynchronous stochastic gradient descent with delay compensation," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Aug. 2017, pp. 4120–4129.

[27] H. Zhu, J. Kuang, M. Yang, and H. Qian, "Client selection with staleness compensation in asynchronous federated learning," *IEEE Trans. Veh. Technol.*, vol. 72, no. 3, pp. 4124–4129, Mar. 2023.

[28] Q. Ma, Y. Xu, H. Xu, Z. Jiang, L. Huang, and H. Huang, "FedSA: A semi-asynchronous federated learning mechanism in heterogeneous edge computing," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3654–3672, Dec. 2021.

[29] Z. Chai et al., "TiFL: A tier-based federated learning system," in *Proc. 29th Int. Symp. High-Perform. Parallel Distrib. Comput.*, New York, NY, USA, 2020, pp. 125–136.

[30] X. Pang et al., "Poisafl: Scalable poisoning attack framework to Byzantine-resilient semi-asynchronous federated learning," in *Proc. 34th USENIX Secur. Symp. (USENIX Secur.)*, Jun. 2025, pp. 6461–6479.

[31] Z. Zheng, Y. Deng, W. Yi, H. Shin, and A. Nallanathan, "Over-the-air computation enabled semi-asynchronous wireless federated learning," *IEEE Trans. Commun.*, vol. 73, no. 10, pp. 8919–8936, Oct. 2025.

[32] Z. Kou, Y. Ji, D. Yang, S. Zhang, and X. Zhong, "Semi-asynchronous over-the-air federated learning over heterogeneous edge devices," *IEEE Trans. Veh. Technol.*, vol. 74, no. 1, pp. 110–125, Jan. 2025.

[33] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, Jul. 2020, pp. 1698–1707.

[34] Y. Liao, Y. Xu, H. Xu, L. Wang, C. Qian, and C. Qiao, "Decentralized federated learning with adaptive configuration for heterogeneous participants," *IEEE Trans. Mobile Comput.*, vol. 23, no. 6, pp. 7453–7469, Jun. 2024.

[35] N. Zhang and M. Tao, "Gradient statistics aware power control for over-the-air federated learning," *IEEE Trans. Wireless Commun.*, vol. 20, no. 8, pp. 5115–5128, Aug. 2021.

[36] Y. Sun, S. Zhou, Z. Niu, and D. Gunduz, "Dynamic scheduling for over-the-air federated edge learning with energy constraints," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 1, pp. 227–242, Jan. 2022.

[37] Y. Shao, D. Gündüz, and S. C. Liew, "Federated edge learning with misaligned over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 21, no. 6, pp. 3951–3964, Jun. 2022.

[38] S. Wang et al., "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[39] Q. Ma, J. Liu, H. Xu, Q. Jia, and R. Xie, "FRACTAL: Data-aware clustering and communication optimization for decentralized federated learning," *IEEE Trans. Big Data*, vol. 11, no. 5, pp. 2102–2118, Oct. 2025.

[40] Y. Xu, Z. Jiang, H. Xu, Z. Wang, C. Qian, and C. Qiao, "Federated learning with client selection and gradient compression in heterogeneous edge systems," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 5446–5461, May 2024.

[41] K. Shen and W. Yu, "Fractional programming for communication systems—Part I: Power control and beamforming," *IEEE Trans. Signal Process.*, vol. 66, no. 10, pp. 2616–2630, May 2018.

[42] W. Dinkelbach, "On nonlinear fractional programming," *Manage. Sci.*, vol. 13, no. 7, pp. 492–498, Mar. 1967.

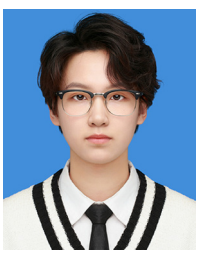
[43] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization," *J. Mach. Learn. Res.*, vol. 17, no. 83, pp. 1–5, 2016.

[44] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[45] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009.

[46] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.
[47] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "Fedbn: Federated learning on non-IID features via local batch normalization," 2021, *arXiv:2102.07623*.
[48] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, Apr. 2011, pp. 1–7.
[49] J. J. Hull, "A database for handwritten text recognition research," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 5, pp. 550–554, May 1994.
[50] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *Proc. Int. Conf. Mach. Learn.*, 2014, pp. 1180–1189.
[51] Q. Ma, Y. Xu, H. Xu, J. Liu, and L. Huang, "FedUC: A unified clustering approach for hierarchical federated learning," *IEEE Trans. Mobile Comput.*, vol. 23, no. 10, pp. 9737–9756, Oct. 2024.
[52] D. W. Hosmer Jr., S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*, vol. 398. Hoboken, NJ, USA: Wiley, 2013.
[53] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory To Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2014.
[54] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[55] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[56] G. Zhu, J. Xu, K. Huang, and S. Cui, "Over-the-air computing for wireless data aggregation in massive IoT," *IEEE Wireless Commun.*, vol. 28, no. 4, pp. 57–65, Aug. 2021.
[57] S. Boyd, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Found. Trends Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2010.

Qianpiao Ma received the B.S. degree in computer science and the Ph.D. degree in computer software and theory from the University of Science and Technology of China, Hefei, China, in 2014 and 2022, respectively. He is currently an Associate Professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China. His primary research interests include federated learning, mobile-edge computing, and distributed machine learning.



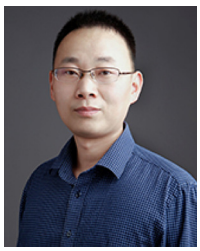
Xiaozhu Song received the B.S. degree in computer science and technology from Nanjing University of Science and Technology, Nanjing, China, in 2024, where she is currently pursuing the M.S. degree. Her research interests include edge computing and federated learning.

Junlong Zhou (Member, IEEE) received the Ph.D. degree in computer science from East China Normal University, Shanghai, China, in 2017. He was a Visiting Scholar with the University of Notre Dame, Notre Dame, IN, USA, from 2014 to 2015. He is currently a Professor with Nanjing University of Science and Technology, Nanjing, China. His research interests include edge computing, cloud computing, and embedded systems, where he has published 120 refereed articles, including more than 40 in premier IEEE/ACM Transactions.

Haibo Wang (Member, IEEE) received the B.E. and master's degrees in computer science from the University of Science and Technology of China in 2016 and 2019, respectively, and the Ph.D. degree in computer science from the University of Florida. He is currently an Assistant Professor with the Department of Computer Science, University of Kentucky. His main research interest include the Internet traffic measurement, big data analytics, software defined networks, and the Internet of Things. His work received IEEE ICNP 2021 Best Paper Award.

Yunming Liao received the B.S. degree from the University of Science and Technology of China, in 2020, where he is currently pursuing the Ph.D. degree with the School of Computer Science and Technology. His research interests include mobile edge computing, federated learning, and edge intelligence.

Jianchun Liu (Member, IEEE) received the Ph.D. degree in computer software and theory from the University of Science and Technology of China in 2022. He is currently an Associate Researcher with the School of Computer Science and Technology, University of Science and Technology of China. His primary research interests include federated learning, mobile edge computing, and distributed machine learning.



Hongli Xu (Member, IEEE) received the Ph.D. degree in computer software and theory from the University of Science and Technology of China, China, in 2007. He is currently a Professor with the School of Computer Science and Technology, University of Science and Technology of China. He has published more than 100 papers in famous journals and conferences. His primary research interest is software defined networks, edge computing, and the Internet of Things. He has won the best paper award or the best paper candidate in several famous conferences.